OLD DOMINION UNIVERSITY RESEARCH FOUNDATION

DEPARTMENT OF MECHANICAL ENGINEERING
SCHOOL OF ENGINEERING
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA 23508

SUPERCOMPUTER IMPLEMENTATION OF FINITE ELEMENT ALGORITHMS FOR HIGH SPEED COMPRESSIBLE FLOWS

By

Earl A. Thornton, Principal Investigator

and

R. Ramakrishnan, Graduate Research Assistant

Progress Report
For the period ended June 30, 1986

Prepared for the
National Aeronautics and Space Administration
Langley Research Center
Hampton, Virginia 23665

Under
Research Grant NSG-1321
Allan R. Weiting, Technical Monitor
LAD-Aerothermal Loads Branch

June 1986

DEPARTMENT OF MECHANICAL ENGINEERING
SCHOOL OF ENGINEERING
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA 23508

SUPERCOMPUTER IMPLEMENTATION OF FINITE ELEMENT ALGORITHMS FOR HIGH SPEED COMPRESSIBLE FLOWS

By

Earl A. Thornton, Principal Investigator

and

R. Ramakrishnan, Graduate Research Assistant

Progress Report
For the period ended June 30, 1986

Prepared for the
National Aeronautics and Space Administration
Langley Research Center
Hampton, Virginia 23665

Under
Research Grant NSG-1321
Allan R. Weiting, Technical Monitor
LAD-Aerothermal Loads Branch

Submitted by the
Old Dominion University Research Foundation
P. O. Box 6369
Norfolk, Virginia 23508

June 1986

SUPERCOMPUTER IMPLEMENTATION OF FINITE ELEMENT ALGORITHMS FOR HIGH SPEED COMPRESSIBLE FLOWS

By

Earl A. Thornton[1] and R. Ramakrishnan[2]

ABSTRACT

Prediction of compressible flow phenomena using the finite element method is of recent origin and considerable interest. Two shock capturing finite element formulations for high speed compressible flows are described. A Taylor-Galerkin formulation uses a Taylor series expansion in time coupled with a Galerkin weighted residual statement. The Taylor-Galerkin algorithms uses explicit artificial dissipation, and the performance of three dissipation models are compared. A Petrov-Galerkin algorithm has as its basis the concepts of streamline upwinding. Vectorization strategies are developed to implement the finite element formulations on the NASA Langley VPS-32. The vectorization scheme results in finite element programs that use vectors of length of the order of the number of nodes or elements. The use of the vectorization procedure speeds up processing rates by over two orders of magnitude. The Taylor-Galerkin and Petrov-Galerkin algorithms are evaluated for 2D inviscid flows on criteria such as solution accuracy, shock resolution, computational speed and storage requirements. The convergence rates for both algorithms are enhanced by local time-stepping schemes. Extension of the vectorization procedure for predicting 2D viscous and 3D inviscid flows are demonstrated. Conclusions are drawn regarding the applicability of the finite element procedures for realistic problems that require hundreds of thousands of nodes.

---

[1]Professor, Department of Mechanical Engineering, Old Dominion University, Norfolk, Virginia 23508.
[2]Graduate Research Assistant, Department of Mechanical Engineering, Old Dominion University, Norfolk, Virginia 23508.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

LIST OF SYMBOLS

A	element area
A_o	coefficient matrix
$\tilde{A}_1$, $\tilde{A}_2$	transformed Jacobian matrices
$\tilde{a}$	characteristic vector
c	local speed of sound
E_i	artificial dissipation flux
E^1, E^2	second and fourth order dissipative fluxes
$e^{(2)}$, $e^{(4)}$	second and fourth order dissipation constants
F, F_i	flux vector
H	entropy function
h	characteristic element length
i	internal energy
k	dissipation coefficient
L	outflow surface length
l_1, l_2, l_3	components of unit normal vector
$[M]$	element mass matrix
M_{ii}	value in global lumped mass matrix for node i
N	element interpolation functions
$\hat{n}$	unit normal vector
p	pressure
Pe	Peclet number

$\{R\}$	load vector, eq. 2.21
S	propagation speed
s	entropy
t	time
Δt	time-step
U	vector of variables
u	typical conservation variable
Δu	incremental conservation variable
u_1, u_2, u_3	velocity components in coordinate directions
V	element volume
W	weighting functions
w	test function, eq. 2.5
x_1, x_2, x_3	coordinate directions
$\Delta x_1, \Delta x_2, \Delta x_3$	increments in coordinate directions
δ_{ij}	Kronecker delta
σ	safety factor for local time-steps
ρ	density
γ	ratio of specific heats
ξ, η	element local coordinates
$\Delta\xi, \Delta\eta$	spatial increments in local coordinates
θ, β	inclination angles of local coordinates

Subscripts

d	constant element quantity
n	time step index
S	surface quantities

s	smoothed quantities
i	component index

Superscripts

n	time step index

Chapter 1

INTRODUCTION

Until the late 19th century scientific analyses were non-existent in engineering practice and engineers relied heavily on empirical relations. The fields of engineering and mathematics were considered to be totally unrelated. The great mathematician Hilbert was said to have pronounced, "The mathematician and the engineer have nothing to do with each other and never will." The historic flight of the Wright brothers in 1903 shattered this notion and ushered in the era of scientific technology wherein research and its practical applications proceeded in parallel. The aerodynamicist of today is very much aware of the strong interplay between mathematics, engineering, and numerical analysis in the prediction of flow behavior around flight vehicles.

A system of nonlinear equations of special interest to aerodynamicists are the compressible flow equations. The compressible inviscid equations or the Euler equations describe flow of a frictionless, non-heat conducting fluid. The addition of viscosity and heat dissipation to the inviscid equations results in the compressible, viscous Navier-Stokes equations. Numerical studies aimed at predicting the flow features these equations describe have mushroomed due to the availability of bigger and better "number crunchers." A new field known as Computational Fluid Dynamics (CFD) has evolved which is devoted to the numerical simulation of fluid dynamic equations.

The role of CFD in fluid dynamics and aerodynamics has seen a dramatic increase in the last few years. A new breed of computational fluid dynamicists are daring to dream of a future free from the tyranny of wind tunnels. The rapid advancements in computer technology with the proliferation of supercomputers and the evolution of ultra-computers indicate that those dreams are fast assuming proportions of reality. The role of CFD has also received a boost with recent initiatives to develop the hypersonic research airplane, the "Orient Express." Numerical simulations will have a major role in the design and development of the hypersonic research aircraft as well as in the design of transatmospheric vehicles (TAVs) that form part of the controversial SDI or "Star Wars" concept.

Supersonic flight vehicles such as the Anglo-French "Concorde" cruise at speeds exceeding Mach 2. The proposed hypersonic research aircraft is envisaged to have applications in the range of Mach 6 to Mach 25. At these extremely high speeds, the aerodynamic heating on vehicle surfaces have substantial effects on flight performance. Deformations and stresses that result from heating effects are significant and means of predicting these effects are clearly necessary. Detailing deformations and stresses due to aerodynamic heating requires integrated fluid-thermal-structural analyses. The accurate prediction of high speed compressible flow features described by the Navier-Stokes equations forms the backbone of such integrated analyses.

1.1 Background

The use of computers to predict flow features has become an important and indispensable part of understanding flow behavior. Use of digital computers to solve problems of fluid dynamics started in the early forties of this century [1]* and has continued ever since. Until the early seventies the method of finite differences enjoyed exclusive use in efforts aimed at predicting flow features. It was only in the early seventies that researchers began to consider the use of finite element methods for flow analyses. Early applications of finite element methods to flow problems were in the incompressible flow domain [2, 3]. During the last three years researchers have developed the first finite element formulations for prediction of high speed compressible inviscid and viscous flows.

A tour d'horizon of research efforts for incompressible and compressible flow simulations using the finite element method appears in [4, 5]. Most of the literature in finite element compressible flows deal with transonic flows. Potential flow formulations have been developed by Ecer and Akay [6] and extensions for the Euler equations in non-conservative form are presented in [7]. The use of implicit and explicit procedures to solve inviscid transonic problems about airfoils and engine inlets is detailed by Argrand, et al. [8, 9]. A Galerkin finite element formulation was recently used by Jameson [10] to study the flow features about an entire Boeing 747 aircraft. Two families of finite element formulations for high speed compressible flow analyses have evolved: the Taylor-Galerkin [11, 13] and the Petrov-Galerkin [14, 16] formulations.

---

*Numbers in brackets indicate references

A question that often arises, especially from devotees of the finite difference method, is why use finite element methods for compressible flow computations? Admittedly, finite difference methods have reached a high level of sophistication, but some capabilities of the finite element method suggest that the method deserves investigation. The need for integrated fluid-thermal-structural analyses was mentioned earlier, and the finite element method lends itself well to such an approach. The capability to model complex flow domains with relative ease is also one of its major selling points. Compressible flow situations are often characterized by regions that need to be adaptively refined. The geometric flexibility of finite elements makes it highly amenable to mesh refinement procedures. Factors such as these have infused new interest in the application of finite element methods to compressible flow problems.

A research effort is underway at the NASA Langley Research Center, in concert with industry and university researchers, to improve the capability and efficiency of finite element methods for high speed compressible flows and to develop more efficient integration of finite element fluid, thermal and structural analyses. The culmination of these research efforts will be the ability to predict accurately aerothermal loads for complex three dimensional bodies.

1.2 Purpose

The need for integrated fluid-thermal-structural analyses and the motivations for choosing the finite element method for such analyses appear in earlier sections. Finite element formulations suitable for

compressible flow calculations are of recent origin. The Taylor-Galerkin and Petrov-Galerkin formulations continue to evolve being modified to enhance their capabilities for accurate solutions. A Flux-Corrected-Transport (FCT) version of the Taylor-Galerkin was developed recently [17], and modifications of the Petrov-Galerkin formulations continue [18].

The procedure to simulate the characteristics of compressible fluid flow involves, in addition to the finite element algorithm, issues such as fast-and-easy model generation, efficient programming strategies to implement the algorithms, and results display using color graphics. In this study the generation of the flow model and the results display are done using the commercially available software package, PATRAN [19].

The use of efficient programming strategies is of major importance to predict flow behavior around complex 3D configurations. The purpose of this study is to develop computational procedures for implementing finite element formulations for compressible flows on a supercomputer. The highly vectorized computational procedures can be used to analyze compressible inviscid and viscous flows and can be extended to develop integrated fluid-thermal-structural analyses.

The basic concepts of shock capturing methods which are based on the theory of weak solutions are introduced in Chap. 2. Two shock capturing finite element formulations are then introduced. The Taylor-Galerkin formulation uses explicit artificial dissipation and three popular dissipation models are explored. The Petrov-Galerkin formulation, being based on the concepts of upwinding, needs no explicit artificial dissipation. Chapter 3 discusses the need for

vectorization strategies and develops procedures to vectorize the finite element algorithms effectively on the NASA Langley VPS-32. The effects of local time-stepping procedures and the use of different explicit dissipation models for the Taylor-Galerkin finite element formulation are illustrated in Chap. 4. The Taylor-Galerkin and Petrov-Galerkin formulations are evaluated for a variety of 2D inviscid problems. Evaluation criteria include solution accuracy, computational speed, and storage requirements. Extension of the vectorization procedure to 2D viscous flows using the Petrov-Galerkin formulation appears in Chap. 5. The finite element methodology for 3D inviscid compressible flow computations using the Taylor-Galerkin formulation is described in Chap. 6. Finally, conclusions are drawn regarding the performance of the two vectorized finite element formulations for inviscid and viscous flows, and recommendations are made for further research.

Chapter 2

SHOCK CAPTURING FINITE ELEMENT ALGORITHMS

Compressible flow problems involve flow situations that may contain shocks, contact surfaces and expansions. Shocks are difficult to model being characterized by abrupt changes in all variables across a very thin region. In reality, the shock occurs across spatial dimensions of the order of microns, but due to computational limitations, shocks are modelled as spread over a few grid points. A numerical scheme is rated according to how well it captures the shock - the crisper the shock, the better the method.

Prediction of shock location and strength can be achieved by either "shock fitting" or "shock capturing." Shock fitting methods use the Rankine-Hugoniot relations to fit the shock relative to the flow field [20]. Shock capturing methods, on the other hand, predict shocks and other discontinuities as part of the solution. Though the predicted shocks are smeared a bit, the generality of the concept makes the method attractive.

2.1 Theory of Weak Solutions

The principles of shock capturing are based on the theory of weak solutions of hyperbolic equations. Consider the hyperbolic system of

equations given by,

$$U_{,t} + F_{,x} = 0 \tag{2.1}$$

where U is a vector of unknowns and is a function of x and t. F is a vector function of U, x and t. The above equation may be written in coefficient form as,

$$U_{,t} + A\, U_{,x} = 0 \tag{2.2}$$

where A, the coefficient matrix, is given by,

$$A = F_{,U} \tag{2.3}$$

Since Eq. (2.1) is a hyperbolic system of equations, the eigenvalues of matrix A are all real.

Nonlinear hyperbolic partial differential equations exhibit two types of solutions, weak and genuine solutions. A genuine solution of the above equations occurs when U is continuous over the domain but derivatives of U may be discontinuous. A weak solution, on the other hand, occurs when U is continuous in the domain, except along a line or surface where U may be discontinuous. The presence of shocks in supersonic flows is an indication of the presence of weak solutions for the hyperbolic equations. Let the solution U(x,t) of Eq. (2.1) be subjected to the initial data,

$$U(x,o) = \phi(x) \tag{2.4}$$

Let w(x,t) be a test function that satisfies the integral version of Eq. (2.1),

$$\iint (w_{,t}\, U + w_{,x}\, F)\, dxdt + \int w(x,0)\, \phi(x)\, dx = 0 \tag{2.5}$$

which is obtained by multiplying Eq. (2.1) by w(x,t) and integrating by parts [21]. If U is a weak solution it can be shown that across a line

of discontinuity,

$$F(U_R) - F(U_L) = S(U_R - U_L) \tag{2.6}$$

where S is the speed of propagation of the discontinuity and U_L and U_R are the states to the left and right of the discontinuity, respectively. For the Euler equations these relations are the shock relations or the "Rankine-Hugoniot" relations.

Weak solutions exhibit nonuniqueness for an initial value problem. For instance, the solution of the Burgers equation,

$$u_{,t} + f_{,x} = 0 \tag{2.7}$$

can be either of two weak solutions. For Eq. (2.7) with initial conditions of

$$u(x,0) = \begin{cases} 0 & x < 0 \\ 1 & x > 0 \end{cases} \tag{2.8}$$

two solutions are possible:

$$u(x,t) = \begin{cases} 0 & x < 0 \\ x/t & 0 < x < t \\ 1 & t < x \end{cases} \tag{2.9}$$

and

$$u(x,t) = \begin{cases} 0 & 2x < t \\ 1 & 2x > t \end{cases} \tag{2.10}$$

This implies that the initial value problem for weak solutions is not a meaningful one. If the mathematical model is to reflect physically relevant solutions, an additional principle is needed to select a unique weak solution. One principle that has found wide acceptance is: "Weak solutions that occur in nature are limits of viscous flow." This

statement forms the basis of the artificial viscosity methods. Adding a diffusion term to Eq. (2.1) results in a nonlinear parabolic system given by,

$$U_{,t} + F_{,x} = \varepsilon\, U_{,xx} \tag{2.11}$$

where ε is the dissipation coefficient. The initial value problem can be solved for a wide variety of initial conditions, and it can be shown that as ε becomes smaller the corresponding solutions converge boundedly to the "right" solution for Eq. (2.1). The addition of this viscosity term does not exactly correspond to the addition of viscosity or heat conduction, but does produce solutions that "make sense." The use of artificial viscosity in simulating compressible flow is common for finite difference methods. The pioneering work in this area was done by Von Neumann and Richtmyer [22] who introduced an artificial dissipation term into the equations so as to give shocks a thickness comparable to the grid spacing. The popular finite difference methods such as those of MacCormack [23], Beam and Warming [24], and Burstein [25] use the concept of artificial viscosity in combination with second order difference schemes.

A number of schemes that are currently popular are those based on upwind differencing. These schemes use the concepts of characteristic theory and wave propagation. They are physically consistent and produce sharp shocks without explicit artificial viscosity. Schemes such as those of Steger and Warming [26], Roe [27], and van Leer [28], are upwind schemes that contain internal dissipation due to the one-sided differencing. Upwind schemes can be shown to be equivalent to central difference schemes with added dissipation [29].

The Taylor-Galerkin formulation which uses the concepts of explicit artificial viscosity, and the Petrov-Galerkin method which is based on the concepts of upwinding are shock-capturing finite element formulations.

2.2 Euler Equations

The conservation equations can be written in two dimensions as,

$$U_{,t} + F_{i,i} = 0 \tag{2.12}$$

where U is a vector of conservation variables, and F_i are flux components of the mass, momentum and energy in the coordinate directions. The index i denotes the component in the i-th direction, and a comma denotes partial differentiation. Repeated indices indicate summation over the range of i. The vector U and the flux vectors F_i are given by,

$$U = \rho \begin{Bmatrix} 1 \\ u_1 \\ u_2 \\ E_t \end{Bmatrix} \qquad F_i = u_i U + p \begin{Bmatrix} 0 \\ \delta_{1i} \\ \delta_{2i} \\ u_i \end{Bmatrix} \tag{2.13}$$

where ρ is the density, u_i are the velocity components in the coordinate directions, and E_t is the total energy. The Kronecker delta δ_{ji} is defined as,

$$\delta_{ji} = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases} \tag{2.14}$$

and the pressure p as,

$$p = (\gamma - 1)\, \rho [E_t - 0.5(u_i u_i)] \tag{2.15}$$

Equation (2.12) is solved subject to proper initial and boundary conditions.

2.3 Taylor-Galerkin Algorithm

The Taylor-Galerkin algorithm was first proposed by Donea for the advection equation [30]. The concept was then applied to the inviscid Euler equations by Lohner, Morgan and Zienkiewicz [11, 12]. One step Taylor-Galerkin and two step Taylor-Galerkin formulations [13] have been applied to high speed inviscid equations.

2.3.1 Finite Element Formulation

The Taylor-Galerkin formulation is easier to derive by considering just one variable. For a scalar variable u the typical equation is,

$$u_{,t} + F_{i,i} = 0 \tag{2.16}$$

where u, F_i are analogus to the corresponding vector quantities in Eq. (2.12). The Taylor-Galerkin formulation used in this dissertation is a two step method with element quantities being calculated at the first step and nodal quantities at the second step. A detailed derivation of the algorithm is presented in [4]; for brevity, only the key equations are presented herein.

Time level $t_{n+1/2}$:

The constant element value $u_d^{n+1/2}$ is computed from,

$$A\, u_d^{n+1/2} = \int_A [N]\, dA\, \{u\}^n - \frac{\Delta t}{2} \int_A [N_{,i}]\, dA\, \{F_i\}^n \tag{2.17}$$

where A denotes an element area, Δt is the time step, and [N] is a row matrix of element interpolation functions. On the outflow plane the element side quantities are computed from,

$$L\, u_s^{n+1/2} = \int_L [N_s]\, ds\, \{u_s\}^n - \frac{\Delta t}{2} \int_L [N_s]\, ds\, \{F_{i,i}\}^n \qquad (2.18)$$

In the above $[N_s]$ denotes the interpolation function of the flux components on the outflow surfaces, and L is the outflow surface length. The interpolation of the flux quantities on the boundary differs from that on the interior. Other variations of calculating the outflow terms, and the rationale behind those calculations appear in [4]. The quantities obtained at the half step are used to calculate the nodal quantities at the second step, $t=t_{n+1}$.

Time level t_{n+1}:

An approximation to the Taylor series expansion of u at $t_{n+i/2}$ and the application of the weighted residual statement on the resulting equations yield,

$$[M]\, \{\delta u\}^{n+1} = \Delta t \int_A [N_{,i}]\, dA\, F_i^{n+1/2} + \{R\}^{n+1/2} \qquad (2.19)$$

where [M] is the element consistent mass matrix given by,

$$[M] = \int_A [N]^T [N]\, dA \qquad (2.20)$$

and the load vector {R} is given by,

$$\{R\}^{n+1/2} = -\Delta t \int_L l_i\, F_{si}^{n+1/2}\, \{N\} ds \qquad (2.21)$$

In Eq. (2.21) l_i are the components of the unit normal surface vector $\hat{n}$. The flux components on the surface F_{si} are computed using the surface quantities obtained at the half step from Eq. (2.18).

The element integrals that appear in Eqs. (2.17)-(2.21) can be evaluated in closed form to avoid numerical integration that is

common in many finite element formulations. Numerical integration, typically Gauss quadrature is expensive and for three dimensions increases computer storage requirements appreciably. The element integrals that appear in Eqs. (2.17)-(2.21) are evaluated just once, outside the time-step loop, making the time-marching scheme very efficient. Reference 31 explains the Taylor-Galerkin formulation used and details the explicit element integral evaluations used in the implementation of the Taylor-Galerkin algorithm.

2.3.2 Artificial Dissipation Models

To guarantee physically consistent results and to stabilize the computations, artificial dissipation is added at the end of each time-step. The a-posteriori smoothing is given by the relation,

$$u_s^{n+1} = u^{n+1} + D(u^{n+1}) \tag{2.22}$$

where D(u) is a diffusion operator that depends on the artificial viscosity model employed. A variety of artificial dissipation models exist, and three popular models are investigated for use with the Taylor-Galerkin algorithm.

Artificial dissipation models that have found wide use in CFD literature include the dissipation models due to Lapidus [32], MacCormack and Baldwin [33], and the blended higher order differencing scheme due to Jameson [34].

2.3.2.1 Lapidus dissipation model: The dissipation operator due to Lapidus [32] is second order and uses velocity gradients as dissipation coefficients. The addition of artificial dissipation is based on the

relation,

$$u_s^{n+1} = u^{n+1} + L\ (u^{n+1}) \tag{2.23}$$

where $u = u(x_i,t)$ is the scalar unknown in Eq. (2.16) and $L(u)$ is the dissipation operator, the dissipation being added at the end of each time-step. The dissipation operator $L(u)$ is defined as,

$$L(u) = E_{i,i} \tag{2.24}$$

where,

$$E_i = k_i\ u_{,i} \qquad \text{(i not summed)} \tag{2.25}$$

The coefficients or pointers, k_i, indicate where and how much dissipation is added in specific regions of the flowfield. The coefficients k_i are given by,

$$k_i = \nu\ \Delta t\ A\left|u_{i,i}\right| \qquad \text{(i not summed)} \tag{2.26}$$

where u_i are the velocity components in the coordinate directions, A is the element area, ν is the Lapidus constant, and Δt is the time-step.

The method of weighted residuals applied to Eq. (2.23) results in the element equation,

$$\int_A [N]^T\ u_s^{n+1}\ dA = \int_A [N]^T\ u^{n+1}\ dA + \int_A [N]^T\ E_{i,i}\ dA \tag{2.27}$$

Using the Green-Gauss theorem for the second order terms, Eq. (2.27) reduces to,

$$\int_A [N]^T\ \Delta u_s^{n+1}\ dA = -\int_A E_i[N,_i]\ dA + \int_L [N]\ E_{i,i}\ ds \tag{2.28}$$

where

$$\Delta u_s = u_s^{n+1} - u^{n+1} \tag{2.29}$$

$$= [N]\ \{\Delta u_s\}^{n+1}$$

The second term in Eq. (2.28) is the boundary term that results due to the integration by parts. The boundary term vanishes at the outflow surfaces, and Eq. (2.28) can be written as,

$$[M]\ \{\Delta u_s\}^{n+1} = -\int_A E_i\ [N,_i]\ dA \tag{2.30}$$

where [M] is the element mass matrix that appears in Eq. (2.20).

2.3.2.2 MacCormack-Baldwin dissipation model: The MacCormack-Baldwin dissipation model [33] uses the second derivative of pressure as a pointer for the amount of artificial dissipation to be added. The addition of artificial diffusion is given by the relation,

$$u_s^{n+1} = u^{n+1} + MB(u^{n+1}) \tag{2.31}$$

where the operator MB(u), the MacCormack-Baldwin dissipation operator is similar in form to the Lapidus dissipation operator and is defined as,

$$MB(u) = E_{i,i} \tag{2.32}$$

The coefficients k_i are given by,

$$k_i = \frac{\nu A^2}{4p} \left| p,_{ii} \right| \qquad (i \text{ not summed}) \tag{2.33}$$

The use of the method of weighted residuals and the Green-Gauss theorem results in the element equation similar to Eq. (2.30) as,

$$[M]\ \{\Delta u_s\}^{n+1} = -\int_A E_i\ [N,_i]\ dA \tag{2.34}$$

where coefficients k_i are given by Eq. (2.33).

The calculation of the coefficients k_i involve obtaining the second derivatives of the pressure with respect to the coordinate

directions. The finite element formulation detailed in the earlier sections assumes a bi-linear variation of the conservation variables within an element. If the pressure is interpolated similarly the second derivatives of the pressure vanish. One way to circumvent this problem is to make use of the Green's formula shown in Appendix A. Appendix A shows that the second derivatives of the pressure at the nodes of an element can be obtained from the equation,

$$[M] \left\{\frac{\partial^2 p}{\partial x^2}\right\} = -\int_A [N_{,x}]\, dA\, p_{,x} \tag{2.35}$$

where, $p_{,x}$, the first derivative of pressure, is computed at the Gauss points. The assemblage of the element quantities Eq. (2.35) and solution of the global equations yields the second derivatives of pressure at the nodes. The procedure for obtaining the second derivatives with respect to the other coordinate direction is similar.

2.3.2.3 Jameson dissipation model: The dissipation model due to Jameson [34] is a blend of second and fourth derivatives with the coefficients depending on the pressure gradients. The second and fourth derivatives are adaptively blended such that at shocks the second derivatives come into play, while in smooth regions the fourth derivative terms are used. The smoothing is done according to the equation,

$$u_s^{n+1} = u^{n+1} + J(u^{n+1}) \tag{2.36}$$

where J(u), the blended smoothing operator, is defined as,

$$J(u) = E^1_{i,i} - E^2_{i,i} \tag{2.37}$$

The dissipative fluxes E_i are given by,

$$E_i^1 = k_i^1 u_{,i} \qquad \text{(i not summed)}$$

$$E_i^2 = k_i^2 u_{,iii} \qquad \text{(i not summed)} \qquad (2.38)$$

The coefficients k_i^1 and k_i^2 are functions of the local pressure gradients and can be written as,

$$k_i^1 = A\, \xi_i^1$$

$$k_i^2 = A^2 \xi_i^2$$

where

$$\xi_i^1 = e^{(2)} A \frac{1}{4p} \left| p_{,ii} \right|$$

$$\xi_i^2 = \max\,(0, e^{(4)} - \xi_i^1) \qquad (2.39)$$

and $e^{(2)}$ and $e^{(4)}$ are constants. Equations (2.37)-(2.39) also indicate that when $e^{(4)}$ is zero, the fourth difference terms vanish resulting in the MacCormack-Baldwin dissipation operator.

The use of the weighted residual formulation and the Green-Gauss theorem yields the finite element equations,

$$[M]\,\{\Delta u_s\}^{n+1} = -\int_A E_i^1\,[N_{,i}]\,dA + \int_A E_i^2\,[N_{,i}]\,dA \qquad (2.40)$$

In addition to the second derivatives of pressure, the Jameson dissipation model requires the computation of the third derivatives of the conservation variables. The second derivatives of the conservation variables are obtained at the nodes of an element as,

$$[M]\,\{U_{,ii}\} = -\int_A [N_{,i}]\,[N]\,dA\,\{U_{,i}\} \qquad \text{(i not summed)} \qquad (2.41)$$

where the first derivatives of the conservation variables are

interpolated linearly within an element. The third derivatives are then obtained from,

$$[M]\{U_{,iii}\} = -\int_A [N_{,i}][N]\,dA\,\{U_{,ii}\} \qquad (i \text{ not summed}) \qquad (2.42)$$

The second derivatives of the pressure are calculated with the first derivatives computed at the Gauss points while the third derivatives of the conservation variables are computed assuming linear variation of the first derivatives within an element.

Numerical results for the Taylor-Galerkin finite element formulation with the Lapidus, MacCormack-Baldwin, and Jameson artificial dissipation models are presented in Chap. 4.

2.4 Petrov-Galerkin Algorithm

The method has as its basis the streamline upwind concepts derived by Hughes and Brooks [35]. The addition of diffusion to stabilize the computation is along streamlines. The directional characteristic of the added diffusion avoids crosswind diffusion. The streamline-upwind Petrov-Galerkin (SUPG) principles were first applied to the linear advection equation and the incompressible Navier-Stokes equations by Hughes and Brooks [36]. Extension of the SUPG concepts to the compressible Euler equations are detailed in [37]. A major drawback of the initial SUPG method was the absence of gradient controls in directions other than the streamline.

2.4.1 Entropy Variables

An enhancement to the SUPG concept was developed by Hughes, et al. [14, 16], which improves the shock capturing capabilities of the

finite element formulation. The enhancements include writing the Euler equations in terms of entropy variables and the use of a "shock-capturing" operator. The use of entropy variables lead to consistent error estimates and also results in a system of symmetric equations which are amenable to error analysis.

The Euler equations given by Eq. (2.12) can be written as,

$$U_{,t} + A_i U_{,i} = 0 \tag{2.43}$$

where A_i are unsymmetric Jacobian matrices in the co-ordinate directions given by,

$$A_i = F_{i,U} \tag{2.44}$$

The form of matrices A_i for the 2D Euler equations appear in Appendix B. The matrices A_i are seen to be unsymmetric. To obtain dimensionally consistent stable results and to obtain entropy conservation from a weak formulation, the Euler equations are symmetrized using entropy functions [38]. A change of variables is introduced by defining new independent variables V to replace the conservation variables U. A one-to-one mapping is assumed between U and V. Equation (2.43) transforms to,

$$A_o V_{,t} + \tilde{A}_i V_{,i} = 0 \tag{2.45}$$

where

$$\begin{aligned} A_o &= U_{,V} \\ \tilde{A}_i &= F_{i,V} = A_i A_o \end{aligned} \tag{2.46}$$

$\tilde{A}_i$ are the transformed Jacobian matrices, see Appendix B. The definition of variables V is such that the transformed Jacobian

matrices are symmetric, and matrix A_0 is positive definite and symmetric. This condition is satisfied by using the generalized entropy function H(U) and defining the variables V as,

$$V^T = H_{,U} \tag{2.47}$$

Hughes et al. [14], use the physical entropy s, in the definition of the entropy function H as,

$$H(U) = -\rho s \tag{2.48}$$

The new variables V, denoted as entropy variables, are thus defined as,

$$V = \frac{1}{\rho T} \begin{Bmatrix} -U_4 + \rho i \ (\gamma + 1 - s) \\ U_2 \\ U_3 \\ -U_1 \end{Bmatrix} \tag{2.49}$$

where s is the entropy given by,

$$s = \ln \frac{(\gamma - 1)\ \rho i}{U_1^{\gamma}} \tag{2.50}$$

and

$$\rho i = U_4 - \frac{U_2^2 + U_3^2}{2U_1} \tag{2.51}$$

The entropy variables and the Jacobian matrices that appear in this dissertation are for 2D inviscid flows. The Euler equations based on entropy functions can be extended directly to 3D inviscid flows, and the coefficient matrices are given in [39].

2.4.2 Finite Element Formulation

The discretization of the domain of interest and application of the method of weighted residuals to Eq. (2.45) results in the relation,

$$\int N^T (A_o V_{,t} + \tilde{A}_i V_{,i}) \, dA = 0 \qquad (2.52)$$

where N is a matrix of element interpolation functions. For the Euler equations, the use of the simple Galerkin (or Bubnov-Galerkin, as it is sometimes referred to) method results in a formulation that conserves entropy. To account for the entropy production at shocks and discontinuities, the Petrov-Galerkin formulation uses weighting functions different from the interpolation functions N.

The weighted residual equations for the Euler equations become,

$$\int W^T (A_o V_{,t} + \tilde{A}_i V_{,i}) \, dA = 0 \qquad (2.53)$$

where W is the weighting function. The definition of W is such that compressible flows with flow characteristics including shocks, contact surfaces and expansions are modelled accurately. The weighting function proposed in [16] contains three components: (a) a streamline operator, (b) a discontinuity capturing operator, and (c) a reduced discontinuity capturing operator. The need for these operators can be illustrated clearly by application to the advection equations. Appendix C details the form of the three operators and the dissipation added by each operator.

For the compressible flow equations the discontinuities are shocks and the three operators can be tagged as the streamline operator, the shock capturing operator and the reduced shock capturing operator. The streamline operator adds diffusion to capture discontinuities and suppress shock related oscillations. The addition of diffusion is directional to avoid overly smoothed results. The directional

characteristics can be obtained from the Jacobian matrices $\tilde{A}_i$ given by Eq. (2.46). The direction of propagation of information can be obtained from a set of real eigenvalues of the Jacobian matrices. The weighting function with the streamline operator can be written as,

$$W = N + S_i^T N_{,i} \tag{2.54}$$

where S_i are the streamline operator matrices given by,

$$S_i = \tilde{A}_i \, T|\Lambda|^{-1} T^T \tag{2.55}$$

where Λ is a set of eigenvalues, and T is a matrix of right eigenvectors.

The use of W defined as in Eq. (2.54) can be used to predict flow features such as shocks, but localized shock related oscillations persist. To suppress these oscillations the Petrov-Galerkin formulation uses the shock capturing operator. The shock capturing operator acts normal to the shock, and this direction is indicated by the gradients of the entropy variables. The definition of the shock capturing operator is based on splitting the Jacobian matrix operator into two parts, a parallel part $\tilde{A}_{i\parallel}$ acting in the direction of the gradient and $\tilde{A}i_{\perp}$, acting normal to the gradient. $\tilde{A}_{i\parallel}$ is defined as,

$$\tilde{A}_{i\parallel}^T \, V_{,i} = \tilde{A}_i^T \, V_{,i}$$

$$\tilde{A}_{i\parallel}^T \, Z_i = 0 \quad \text{for } Z_i \perp V_{,i} \tag{2.56}$$

where Z_i are vectors normal to the direction of discontinuity. The use of the parallel components $\tilde{A}_{i\parallel}$ in the formulation modifies the weighting function and W can be written as,

$$W = N + S_i^T N_{,i} + R_i^T N_{,i} \tag{2.57}$$

where R_i are the shock capturing operator matrices. The matrices R_i are given by,

$$R_i = \tilde{A}_{i\|} \; T_\| \; |\Lambda_\||^{-1} \; T_\|^T \tag{2.58}$$

$\Lambda_\|$ is the vector of eigenvalues, and $T_\|$ is the matrix of right eigenvectors of the transformed matrices $\tilde{A}_{i\|}$.

Thus the Petrov-Galerkin formulation has two components - a generalized streamline operator and a shock capturing operator. The explicit control of the gradients is obtained by the shock-capturing term which implies that the component of the streamline operator in the gradient direction is redundant. The projection of the streamline operator in the direction of the gradient can be subtracted out. The "reduced shock capturing" term is given by,

$$Y_i = \tilde{A}_i \; P \tag{2.59}$$

where P is the projection operator which can be written as

$$P = T_\| \; \alpha \; T_\|^T \tag{2.60}$$

α is a scalar, and $T_\|$ is the matrix of eigenvectors that appears in Eq. (2.58). With the inclusion of the reduced shock capturing operator the weighting functions W become,

$$W = N + S_i^T N_{,i} + R_i^T N_{,i} + Y_i^T N_{,i} \tag{2.61}$$

and the weighted residual formulation is written as,

$$\int_A W^T \; (A_o \; V_{,t} + \tilde{A}_i \; V_{,i}) \; dA = 0 \tag{2.62}$$

Details of the derivation of the coefficient matrices and the definition of the shock capturing operators appear in [39].

The finite element formulations described are designed to be pseudo-time marching schemes. The intent is to study problems which could be marched out in time to steady state. This suggests an approximation of the transient terms in Eq. (2.62) as,

$$\int_A W^T A_o V_{,t} \, dA = \int_A N^T A_o V_{,t} \, dA \tag{2.63}$$

Equation (2.62) can then be written as,

$$\int_A N^T A_o V_{,t} \, dA = -\int_A W^T \tilde{A}_i V_{,i} \, dA \tag{2.64}$$

During the transient procedure, matrix A_o which appears on the LHS of Eq. (2.64) is computed at nodes. This makes possible the solution of the global equations given by,

$$A_o \int_A N^T N \, dA \{V_{,t}\} = -\int_A W^T \tilde{A}_i V_{,i} \, dA \tag{2.65}$$

which can be rewritten as,

$$A_o \quad M_{ii} (V_i)_{,t} = f_i \qquad i = 1,2...\text{nodes} \tag{2.66}$$

Equation (2.65) is valid for each node in the domain. M_{ii} is the value in the global mass matrix corresponding to node i, A_o is the coefficient matrix computed at node i, and f_i is the corresponding right hand side value. The time stepping procedure that results is given by,

$$V_i^{n+1} = V_i^n + \frac{\Delta t \, A_o^{-1} f_i}{M_{ii}} \qquad i = 1,2,3...\text{nodes} \tag{2.67}$$

where V_i^{n+1} is the value of the entropy variable at node i at

$t=t_{n+1}$, V_i^n the nodal value at $t= t_n$ and Δt the time-step. Details of transient algorithm used in this study can be found in [39].

2.5 Comments on finite element algorithms

The previous sections describe the formulations for the Taylor-Galerkin and Petrov-Galerkin finite element algorithms. A brief overview contrasting the characteristics of the two algorithms is appropriate.

Both the Taylor-Galerkin and the Petrov-Galerkin algorithms are based on the principle of shock capturing. They differ by virtue of how the necessary artificial dissipation is added. The Taylor-Galerkin uses explicit artificial dissipation to capture shocks while the Petrov-Galerkin uses implicit numerical dissipation. To simplify the solution procedure both algorithms use explicit time-stepping schemes.

The Taylor-Galerkin formulation based on Taylor series expansion is a model of algorithm simplicity. In contrast, the Petrov-Galerkin formulation is rather complicated. The definition of the weighting functions is complex and involves numerous matrix multiplications. Matrix multiplications expand memory requirements and are computationally expensive. Algorithms are usually reformulated to avoid such operations. The evolution of the one step Taylor-Galerkin to the two step formulation used in this dissertation is typical of this desire to eschew matrix multiplications.

The implementation of the Taylor-Galerkin algorithm is simpler since the element integrals that appear in the formulation can be

evaluated in closed form. This procedure can be done just once outside the time stepping procedure. The Petrov-Galerkin algorithm yields non-linear element integrals which need to be evaluated using numerical integration procedures for each time-step.

The Petrov-Galerkin formulation has good mathematical features since it is conducive to accurate error analysis and error estimates, and in the limit of vanishing dissipation ensures conservation of entropy. The algorithm is based on "upwinding" and needs no "tuning" parameters. Reduced integration procedures on the Petrov-Galerkin formulation have shown encouraging trends and may result in reduced computational expense with accuracy of results comparable to full integration procedures.

The primary evaluation criteria for both algorithms include solution quality, shock resolution, extent of vectorization, computational speed, and ease of extension to multidimensional inviscid and viscous flows. The chapters that follow develop vectorization strategies for the two formulations and demonstrate the performance of the algorithms for inviscid and viscous flows.

Chapter 3

VECTORIZATION STRATEGIES FOR FINITE ELEMENT CFD

To predict flows encountered in realistic problems accurately, two ingredients are essential. The first is the development of finite element methodologies such as the Taylor-Galerkin and Petrov-Galerkin, described in the previous chapter. The second ingredient is the effective implementation of the finite element formulation on the computational facility at hand.

For realistic 3D problems the number of degrees of freedom needed for an accurate analysis is astronomical, and the size of the database that must be handled taxes the storage available on most computers. Supercomputers, with their huge central memory as well as large and fast secondary memory, are designed to address such storage demands. Driven by the insatiable needs of computational fluid dynamicists for more memory and higher computational speeds, the supercomputer has in a very short time become an essential research tool for serious CFD researchers.

On a supercomputer, the desirability for low computational costs and rapid turn-around times dictate the need for very effective programming strategies, in particular strategies aimed at vectorizing the computational procedure. Supercomputers, such as the CRAY II, CDC 205 and the Langley VPS-32, are all endowed with large central memory but differ in their hardware configuration. Vectorization strategies

employed on these computers should be tailored to fit each machine's special characteristics if optimum performance is to be approached.

The vectorization procedure should also be global, that is, as much of the program as possible should be vectorized. Figure 3.1 shows the ratio of computational speed to maximum computational speed attainable for various level of vectorization. The three curves correspond to computers with vector/scalar speeds of 5, 10, and 20. It can be seen for the VPS-32 (vector/scalar ratio of 20), a program that is 90% vectorized will run only at 30% of the maximum attainable speed. Vectorization levels close to 100% are a must if the intent is to work problems modelled with elements and nodes that number in the hundreds of thousands.

3.1 VPS-32 Characteristics

The NASA Langley Research Center uses the VPS-32, a CYBER 205 with a central memory of 32 million full precision words, for its CFD applications. The special features of the VPS-32 which set it apart from scalar machines are its virtual memory and pipeline processing capability.

Virtual memory: The VPS-32 is a virtual memory computer which means that the user has the ability to access a virtual address range that exceeds the physical size of the central memory. Information not within the central memory is brought into central memory by the operating system. Information, which may be working arrays or code, resides on blocks of data called pages. The process of retrieving pages, placing them in central memory, and if needed, removing other pages

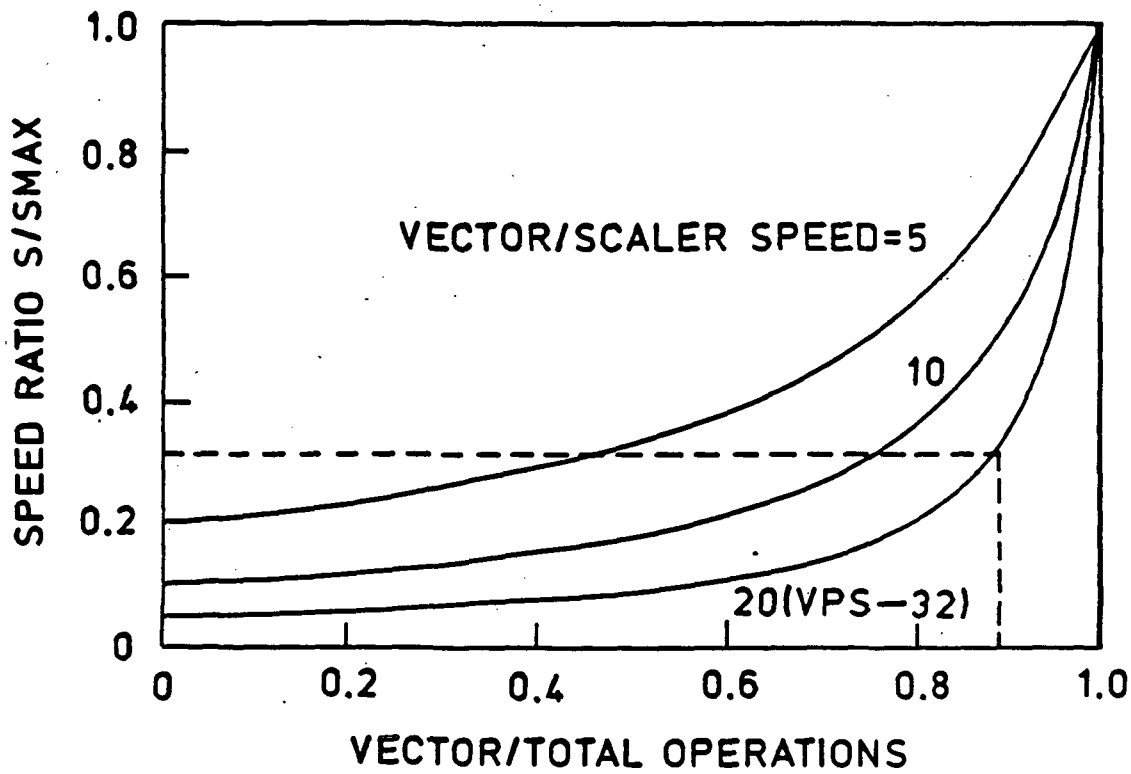


Fig. 3.1 Ratio of computational speed to extent of program vectorization.

from the central memory is called paging. Virtual memory is useful in cases where a "small" problem is made "big" by change of one or more parameters.

Pipeline processing: One of the principal features of the VPS-32 is its efficiency in handling long vectors. The reason for this efficiency is the concept of "pipeline processing." The arithmetic hardware consists of two units or pipelines, Pipe 1 and Pipe 2. Pipe 1 is used for all vector arithmetic operations except divide and square root. Pipe 2 is used for all vector operations. Each pipeline is segmented, that is, a portion of the specific operation on two operands is done at the same time at the first step. The results of the first step are moved down to the next step of the operation while a new set of operands is moved into the first step. The segmented or pipeline construction of the vector machine allows vectors to be streamed through the pipeline at high rates. A good measure of effective vectorization on the VPS-32 is obtained by looking at the timing information for various operations. The vector timing for operands of length n is given by,

$$T = S + n/\ell$$

where T is the time in seconds, S the startup time, and ℓ the number of results obtained per operation. For example, the multiplication operation has the timing information given by,

$$T = 52 + n/2$$

where ℓ is taken to be 2 for the VPS-32.

It is seen that for a fixed number of operations, the longer the vector the more efficient the computational procedure. This is due to

the fact that each vector operation requires a startup time or overhead time to begin operation, but after the first result the succeeding results occur at very high rates. The most effective rates are when n is very large and close to optimum rates are achieved for vector lengths in excess of a thousand.

3.2 Strategies for Implementation of Finite Element CFD

The characteristics of the VPS-32 indicate that a finite element procedure should be implemented such that most, if not all, operations are done with long vectors. Vectorization procedures for implicit algorithms designed to take advantage of this concept are detailed by Noor and Lambiotte [40]. The procedure advocated in [40] works well for dynamic analysis of structures using higher order finite elements. The vectorization procedure takes advantage of the large number of nodes per element and the need for a large number of integration points. This procedure is not well suited for fluid flows. Finite element compressible flow programs use simple elements (3 or 4 nodes/element in 2D) and numerical integration of order 2 is usually adequate.

Finite element procedures for simulating compressible flow features usually employ explicit time-stepping procedures. The explicit solution of the global equations is based by computing element matrices that occur in equations such as Eqs. (2.19) and (2.64). The left-hand and right-hand sides of Eqs. (2.19) and (2.64) are assembled resulting in a global system of equations given by Eq. (2.67). The right hand side vectors on an element level are called element residual vectors,

and the assemblage of the element vector results in the global residual vector. The solution of the system equations, Eq. (2.67) is repeated for each time-step until the conservation variables are perceived to have attained steady-state.

Typical finite element formulations contain processes that are element-to-node operations and node-to-element operations which are not easily vectorizable. These hard-to-vectorize processes include:

(a) element localization
(b) computational of element residuals
(c) assembly of global residual vectors
(d) solution of the global equations
(e) application of boundary conditions
(f) Gauss integration

Tasks (a) to (f) are all critical since they are repeated at each timestep. Effective processing rates are obtained only when all these operations are highly vectorized. The vectorizing strategies that follow were developed to exploit the hardware and software capabilities of the VPS-32. However, the overall vectorization strategy can be used to implement finite element formulations effectively for compressible flow analyses on other supercomputers such as CRAY machines.

3.2.1 Element Localization

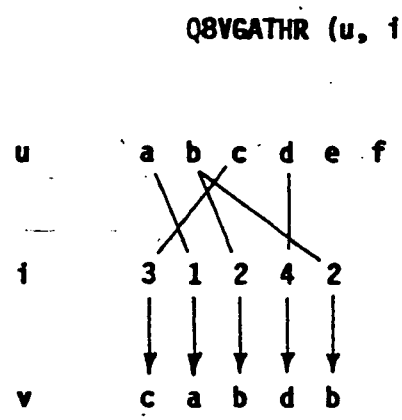
In finite element procedures the region defined by the flow field is discretized into elements. The node numbers associated with an

element are contained in a row of the element connectivity matrix. The coordinates of the nodes defining an element, the value of the variables at these nodes, and other characteristic element quantities are obtained from global arrays using the connectivity row. This process is termed as localization. The vectorization of the localization process is simplified by the use of the Fortran supplied "gather" function on the VPS-32. The gather function generates a vector of real or integer numbers from a vector of the same type using an integer index vector. The use of the "gather" is illustrated in Fig. 3.2(a). The index vector i is used to gather appropriate values of vector u into the vector v.

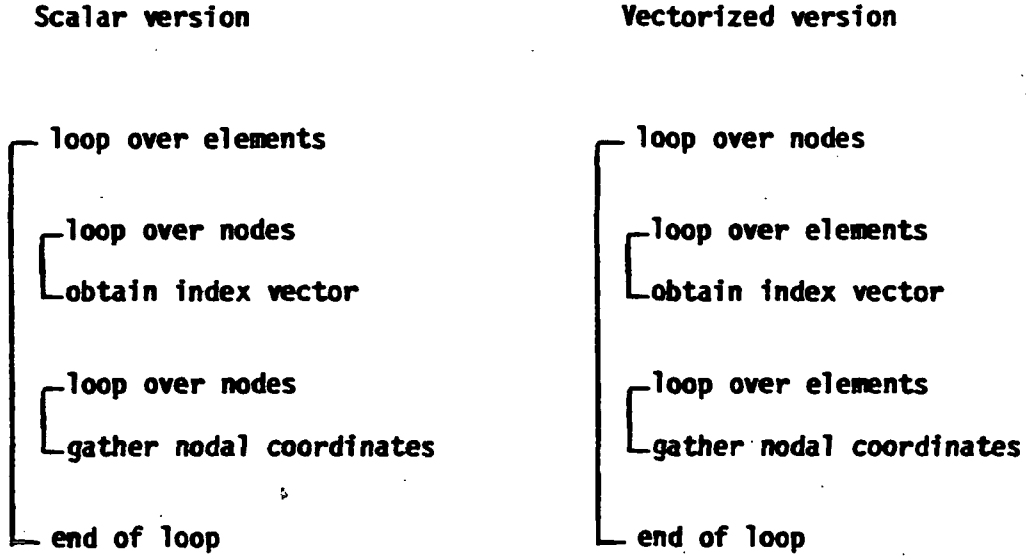
The program structure of the scalar and vectorized versions for the process of obtaining element nodal coordinates is shown in Fig. 3.2(b). The element connectivity matrix plays the role of index vectors to obtain the "right" nodes for the localization process. For the scalar version, the index vector is a row of the connectivity matrix which is of length equal to the number of nodes per element. The index vector on the vectorized version is a column of the connectivity matrix which is of length equal to the number of elements in the discretized domain. Since realistic problems use elements that number in the thousands, the vectorized version of the localization procedure is highly efficient. The procedure to obtain other element quantities, such as element nodal variables, is vectorized along similar lines.

3.2.2 Computation of Element Residuals

Finite element scalar codes generate the global residual vector by the following sequence of operations: (1) evaluate the stiffness



(a) Example of a gather operation



(b) Localization of nodal coordinates

Fig. 3.2 Use of the gather function to obtain nodal coordinates of elements.

matrix for an element, (2) compute the element residual matrix using the vector of element nodal variables, (3) assemble the element residual matrix into the global residual vector using the connectivity array, and (4) repeat steps (1) to (3) for each element. For most finite element programs, the generation of the element residual matrices and the assemblage of these residual matrices into the global residual vector lays a heavy demand on the computational resources available.

Generation of the element stiffness matrices is vectorized by using the vectors obtained from the localization process. The arrays that result from vectorization of the localization process of 3.2.1, are of length equal to the number of elements in the discretized domain, and use of these vectors enables vectorization of the element stiffness computations. To gain further insight into how the scalar and vectorized versions for generating the element stiffness matrices differ, consider the "chips and soda straws" analogy shown in Fig. 3.3. The scalar version bears resemblance to the stack perceived in a can of Pringles potato chips. The ordering is horizontal and each chip can be considered as an element stiffness matrix. However, the vectorized version generates the long vectors and the ordering is vertical, reminiscent of a stack of soda straws. Element stiffness matrices generated by this procedure are used to compute the residual matrices for all elements.

The assembly of the element residual matrices into a global residual matrix is vectorized with the aid of the "scatter" function available on the VPS-32. The scatter operation is the reverse of the gather

ELEMENT STIFFNESS MATRICES

NUMBER OF ELEMENTS = N

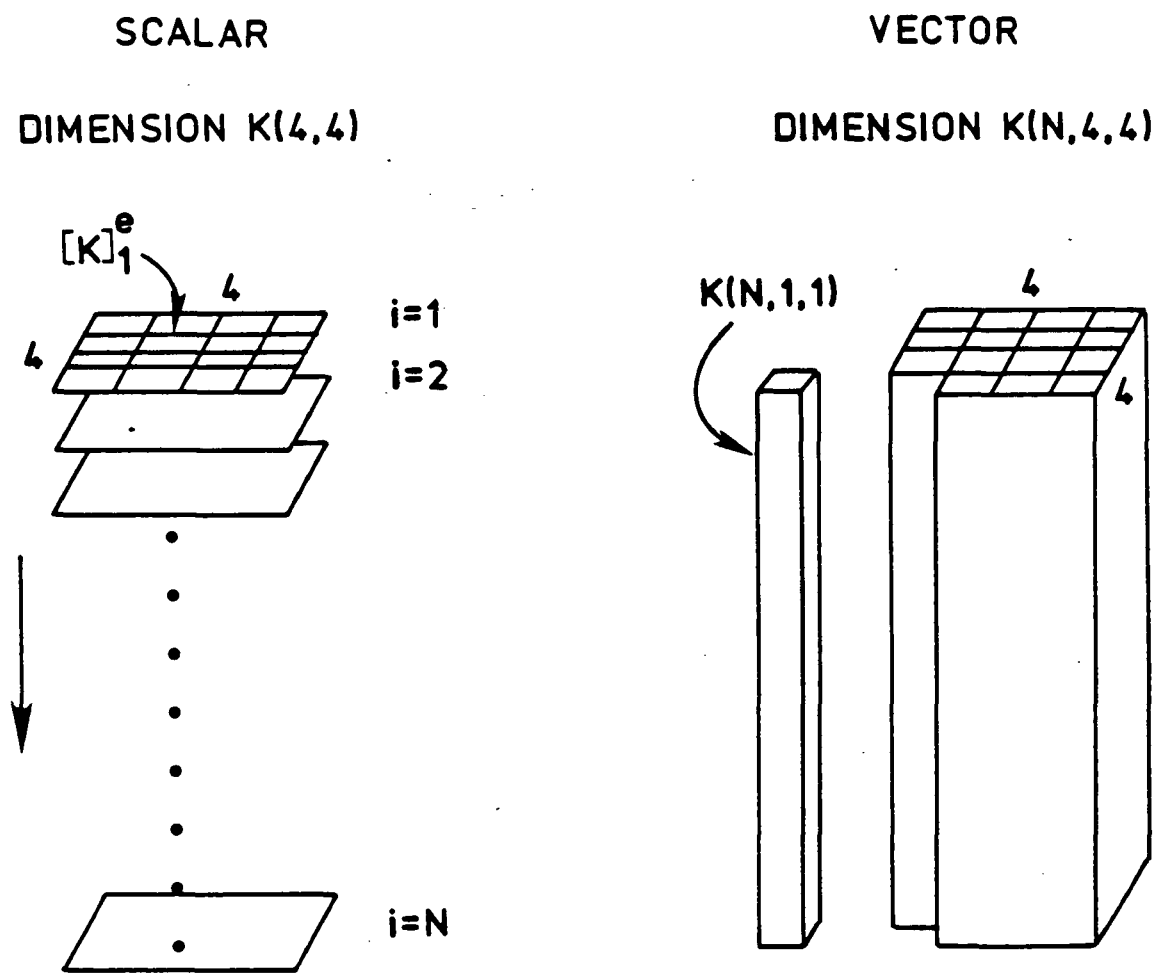


Fig. 3.3 Illustration of differences in scalar and vectorized versions for generating element matrices.

operation and is illustrated in Fig. 3.4. The index vector "i" is used to scatter the appropriate values of vector u into the vector v. It is to be noted that if the index vector has repeating numbers in it, the value indicated by the first index is overwritten by that pointed to by the second repeated index. In the example in Fig. 3.4 "a" was overwritten by "d" as the first value in vector v.

3.2.3 Assembly of Element Residual Vectors

Finite element meshes can be either "structured" or "unstructured." A structured mesh results when the node numbering in a mesh is "well ordered" and the region modelled is of simple geometry. A structured mesh and its connectivity appear in Fig. 3.5. The node numbering for the finite element mesh is such that the connectivity starts at the node on the lower left hand corner of the element and goes counterclockwise. The vectorized version, as indicated earlier, uses the columns of the connectivity matrix as index vectors for the assemblage process. For a structured mesh, a node number appears but once in the same column, which reduces the assemblage to simple scatter operations.

Typical finite element meshes are "unstructured." This implies that the number of elements connected to a node is not constant within the domain. An unstructured mesh and its connectivity appear in Fig. 3.6. It is seen that in column 3 of the connectivity matrix node 11 appears twice. Using a simple scatter with this column as the index vector will produce erroneous results. This is due to the fact that the value pointed to by the second index in that column is overwritten by the value pointed to by the fifth index in the same column. This

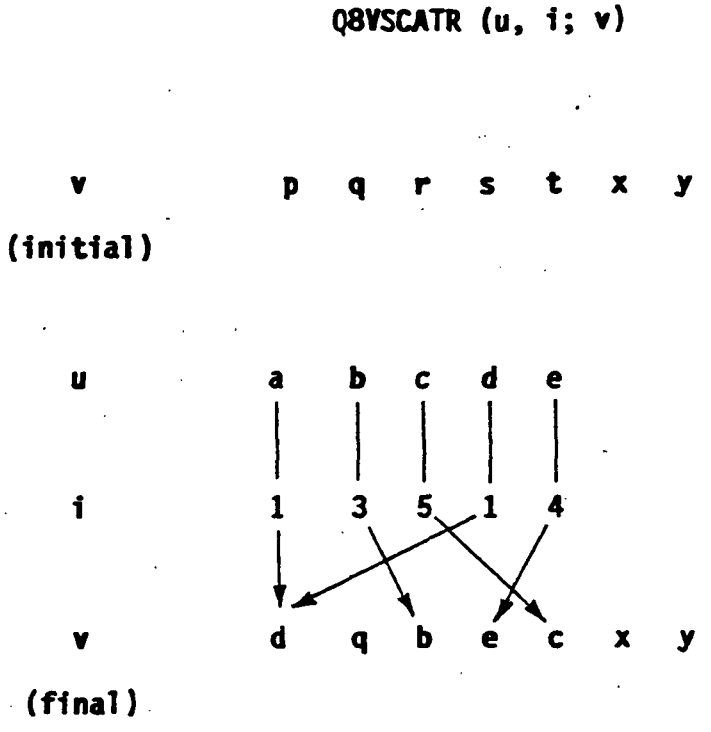
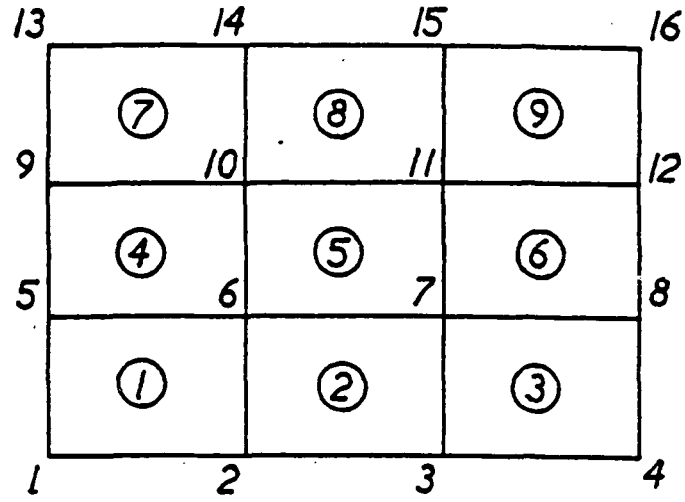
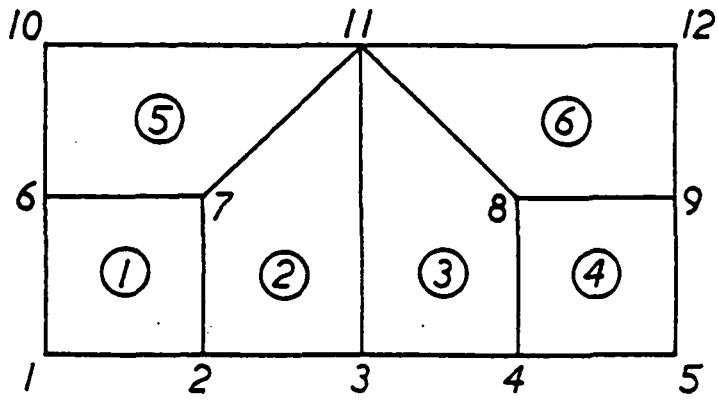


Fig. 3.4 Illustration of the scatter operation.



Element Number	Nodes			
	i	j	k	l
1	1	2	6	5
2	2	3	7	6
3	3	4	8	7
4	5	6	10	9
5	6	7	11	10
6	7	8	12	11
7	9	10	14	13
8	10	11	15	14
9	11	12	16	15

Fig. 3.5 Structured finite element mesh and connectivity matrix.



(a) Finite element unstructured mesh

Element Number	Connectivity				Events counter			
	i	j	k	l	i	j	k	l
1	1	2	7	6	1	1	1	1
2	2	3	11	7	1	1	1	1
3	3	4	8	11	1	1	1	1
4	4	5	9	8	1	1	1	1
5	6	7	11	10	1	1	2	1
6	8	9	12	11	1	1	1	2

(b) Connectivity array (c) Events counter array

Fig. 3.6 Unstructured finite element mesh and associated matrices.

loss of information can be prevented by the use of a "recursive-scatter."

The recursive-scatter is a multipass scatter, the number of passes needed being equal to the number of times a node is repeated in a column of the connectivity array. The number of times a node appears in a column is tagged by an integer array, denoted as the "events counter" array. When a node repeats twice a value of 2 is entered in the events counter array corresponding to the location of that node in the connectivity array. The scattering process uses the events counter array to ensure that no information is lost or overwritten. The events counter array for the unstructured mesh in Fig. 3.6a appears in Fig. 3.6b. The assemblage procedure using the recursive-scatter concept is detailed in Fig. 3.7.

The assembly process uses columns of the residual arrays, the length of a vector being equal to the number of elements in the domain. For the scatter process, columns of the connectivity array serve as index vectors. Node 11 appears in column 3 twice and two passes or iterations will have to be made to properly assemble the residual matrices corresponding to this column of the connectivity matrix. At the first pass the first occurrences of all the nodes are used as pointers to assemble element residual vectors. During this first pass, the second occurrence of nodes are assigned to point to a dummy location. On the second pass, the second occurrence of nodes are used as pointers for assemblage while the first occurrences are dumped into the dummy location. For an assembly process that needs two passes, such as for the mesh on Fig. 3.6, the results obtained from the two passes are added. The event counter matrix thus permits the use of

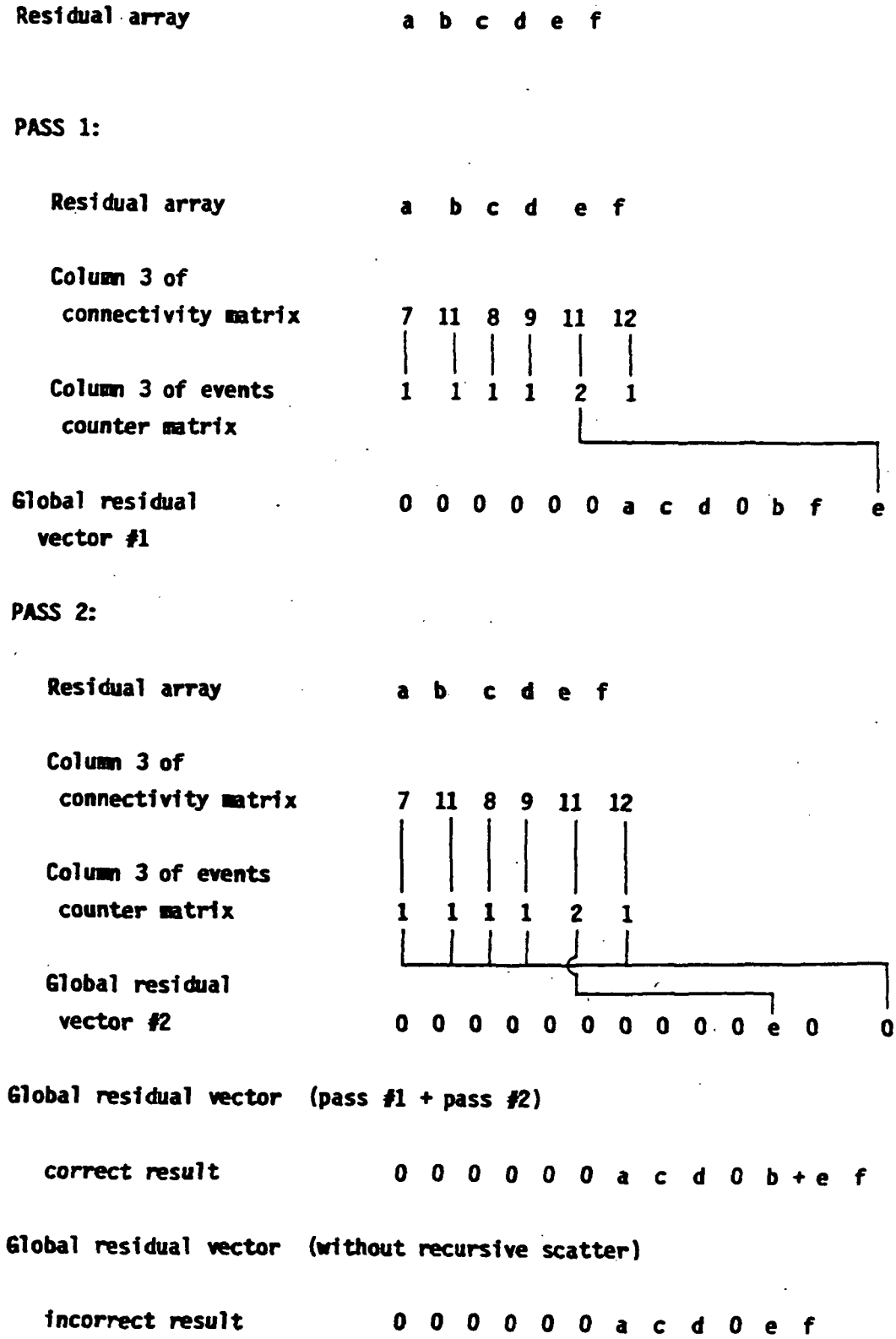


Fig. 3.7 Assembly process using recursive scatter.

an unstructured mesh but makes sure that the assemblage process is done consistently. Figure 3.7 also shows the erroneous results that might accrue using a simple scatter on an unstructured mesh.

The section of the code which computes the events counter array is not vectorized. The penalty to be paid for this scalar operation is quite small since the generation of the events counter array is done outside the time loop. The use of the events counter array enables the assemblage procedure, which has to be repeated for each time-step, to be fully vectorized. The scalar and vectorized versions of the process of generating element stiffness matrices and their assemblage appears in Fig. 3.8.

3.2.4 Solution of Global Equations

Element equations, such as Eqs. (2.19) and (2.64), are typically evaluated for each element and then assembled into global arrays according to the locations defined by the connectivity array. The assembled equations are of the form

$$M_{ii}\ \delta U_i = R_i \tag{3.1}$$

where M_{ii} is the element in the global lumped mass matrix corresponding to node i, δU_i is the solution vector corresponding to node i, and R_i is the corresponding nodal residual. The element lumped mass matrix is calculated using closed form integration and then assembled to form the left hand side of Eq. (3.1). The assembly process is done using the events counter array for unstructured meshes.

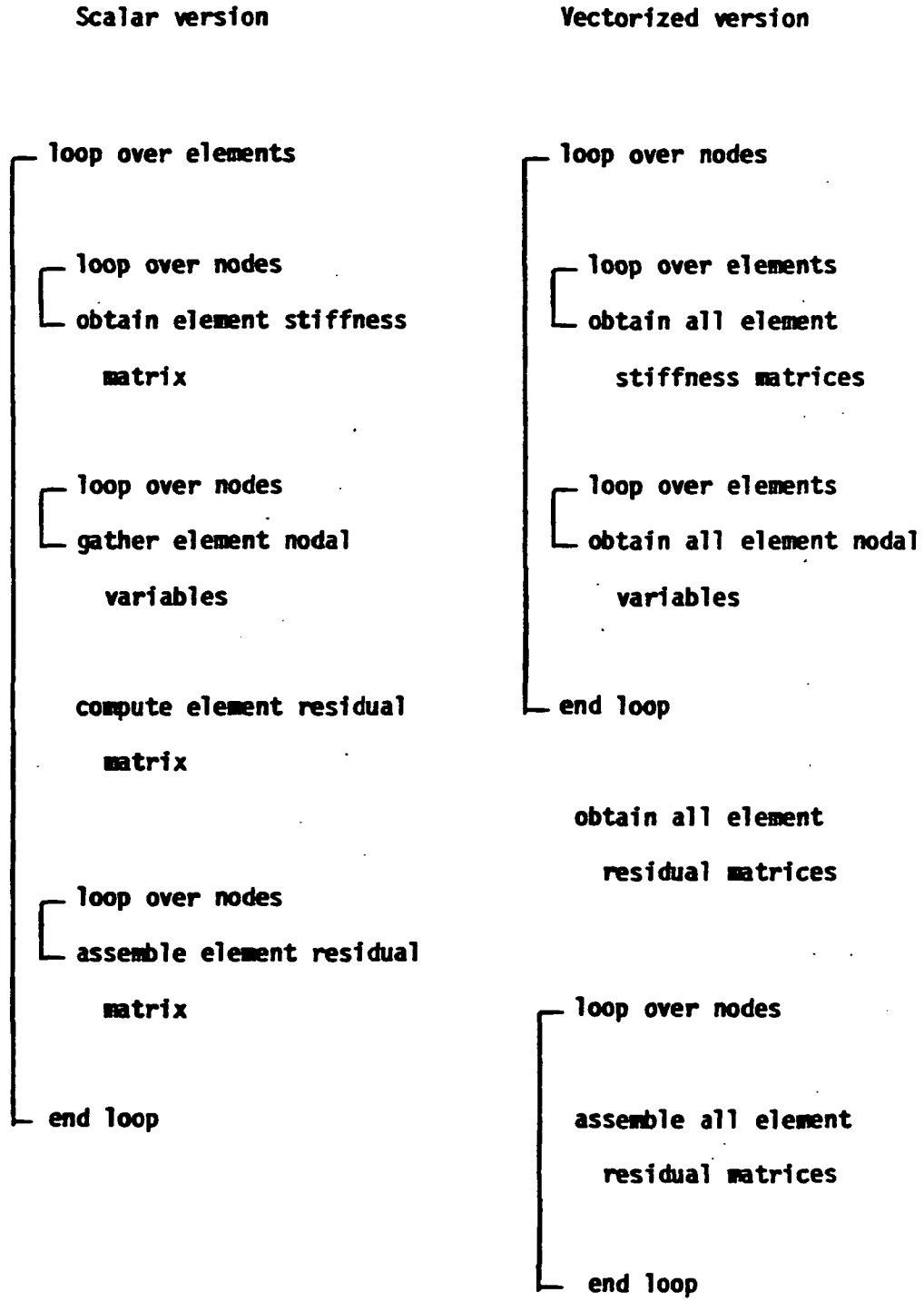


Fig. 3.8 Scalar and vectorized versions for generating global residuals.

A multipass iterative scheme can be used to include the contributions of the non-diagonal terms in the consistent mass matrix [4]. The inclusion of the non-diagonal terms improves solution quality for structural analysis, but this trend has not been established for compressible flow calculations.

The use of vectors of length equal to the number of elements in the domain, and the use of the scatter, or the recursive-scatter if need be, ensure the procedure of calculating the global mass matrix to be fully vectorized. The solution of the system equations is then a simple vector division operation.

3.2.5 Application of Boundary Conditions

Inviscid analysis of flow fields include processing nodes that may have constraints such as slip and no penetration. The vectorization procedure uses gather operation to grab specific nodes, processes these nodes, and then scatters them back into the global arrays. Since the solution process is explicit, the imposition of the boundary conditions is done after the solution of the global equations. Use of gather and scatter functions ensure effective vectorization of this process.

3.2.6 Gauss Integration

Element integrals that are complex or those that contain nonlinear quantities cannot be evaluated in closed form and must be evaluated using numerical integration techniques. The Taylor-Galerkin artificial dissipation terms that appear in Eqs. (2.30), (2.34) and (2.40), and the Petrov-Galerkin element integrals in Eq. (2.64) are evaluated using

numerical integration. The numerical evaluation of these integrals replaces a typical integral by a summation process. The most popular evaluation procedure for finite element integrals is Guass-Legendre quadrature [41]. The number of sampling points used for the element evaluation depends on the degree of the polynomial in the integral. If the element integrals involve complex expressions, numerical experiments may have to be done to determine the number of integration points needed for sufficient accuracy.

To illustrate the strategy to effectively vectorize the process of numerical integration, the generation of a global stiffness matrix is explained. For the scalar version the process is to: (a) evaluate the contribution of the element stiffness matrix due to one Gauss point, (b) add up the contributions to a element stiffness matrix from all the Gauss points, (c) assemble the element stiffness matrix into the global stiffness matrix, and (d) repeat this procedure for each element in the domain.

The vectorized version of this procedure turns the scalar concept inside out. The inner loop, the loop over the Gauss points, becomes the outer loop for the vectorized version. The flowchart for the two versions is illustrated in Fig. 3.9. The vector lengths used in the numerical integration is of the order of the number of elements in the domain which implies good vectorization.

3.3 Comments on Vectorization Strategies

This chapter introduced strategies for vectorizing explicit time-marching finite element algorithms for compressible flow. The

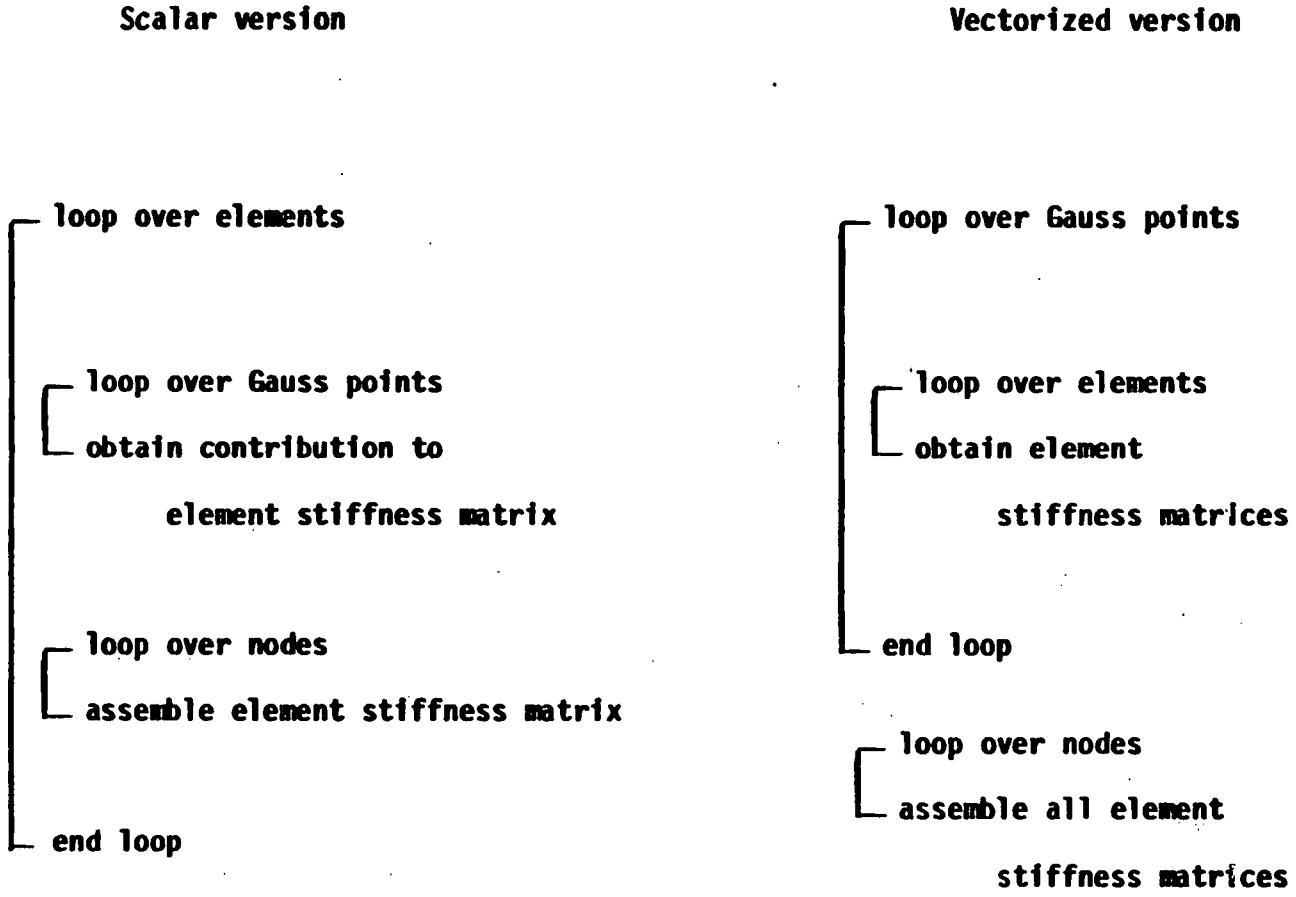


Fig. 3.9 Program structure for scalar and vectorized versions for integral evaluations using Gauss quadrature.

procedures described were used to develop vectorized Taylor-Galerkin and Petrov-Galerkin programs. The programs developed include 2D inviscid, 2D viscous and 3D inviscid flow codes. In the following chapters these finite element programs are used to detail inviscid and viscous features for a variety of flow problems. The results obtained from these analyses are used to compare and contrast the performance of the Taylor-Galerkin and Petrov-Galerkin algorithms.

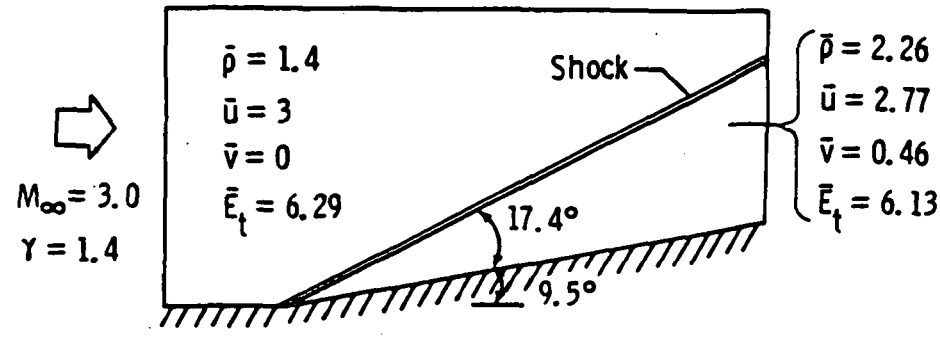
Chapter 4

COMPUTATIONS FOR 2D INVISCID FLOWS

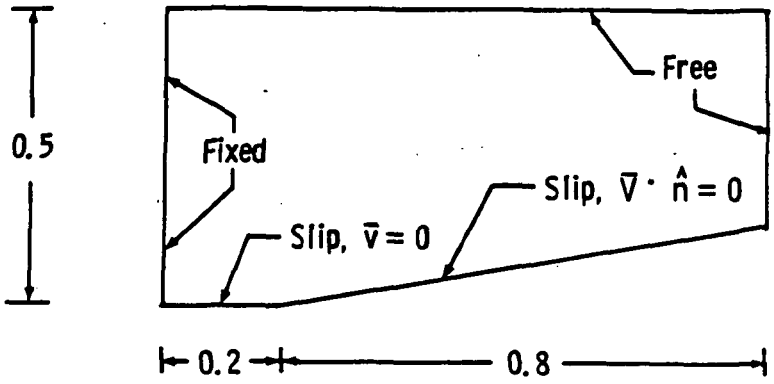
The Taylor-Galerkin and Petrov-Galerkin finite element formulations described in Chap. 2 were applied to 2D inviscid flow problems to assess solution accuracy, shock resolution, convergence rates, and computational speed. The Taylor-Galerkin formulation uses explicit artificial dissipation and the performance of the three dissipation models introduced in Chap. 2 are compared. The use of local time-stepping schemes to stimulate faster convergence rates is also investigated.

4.1 Performance of Artificial Dissipation Models

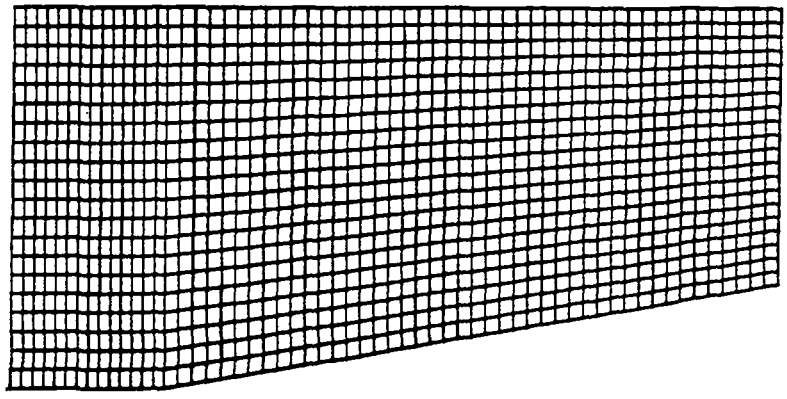
The performance of the three dissipation models for the Taylor-Galerkin algorithm are compared by solving a flow problem with an exact solution. Mach 3 inviscid flow over a compression corner is chosen for this purpose. The flow parameters after the shock can be computed exactly by using oblique shock relations. The flow variables for the entire flow field for the compression corner are shown in Fig. 4.1a. The top boundary is assumed free, and slip boundary conditions are imposed along the wall as shown in Fig. 4.1b. The finite element mesh used for comparing the performance of the three dissipation models contains 1239 nodes and 1160 elements, Fig. 4.1c.



(a) Flow variables (exact solutions)



(b) Boundary conditions



(c) Finite element mesh

Fig. 4.1 Flow configuration and finite element mesh for compression corner.

The density contours obtained using the Lapidus dissipation model appear in Fig. 4.2. A dissipation constant of $\nu = 1$ was used for the computations. The contours show the presence of spurious oscillations at the root of the oblique shock, and an inflow of spurious information from outside the domain is seen at the free top boundary. The freestream oscillations show an undershoot of 7% (1.23 instead of 1.4).

The density contours obtained using the MacCormack-Baldwin dissipation model are shown in Fig. 4.3. The use of the MacCormack-Baldwin dissipation model results in a dramatic reduction in the freestream oscillations. The results also indicate the absence of the spurious oscillations at the top of the region. The shock structure is seen to be crisper with the contour lines coming together at the shock. A small undershoot of about 2% is seen close to the shock.

The density contours obtained using the Jameson dissipation model with dissipation coefficients of $e^{(2)} = 1$ and $e^{(4)} = 1/64$ are shown in Fig. 4.4. The blended dissipation model of Jameson yields results that show lesser oscillations than the Lapidus model, and the freestream oscillations are limited to less than 1% (1.39 instead of 1.4). The contours in Fig. 4.4 also indicate the inflow of spurious oscillations from outside the domain at the top.

A further comparison of the solutions obtained by the use of the three dissipation models is shown in Fig 4.5. The density distribution at the outflow shows the superior results obtained by the MacCormack-Baldwin and Jameson dissipation models. The results obtained by the Jameson model also show better control of post-shock freestream oscillations. The Lapidus dissipation model shows the presence of shock related oscillations and the overall solution accuracy is quite poor.

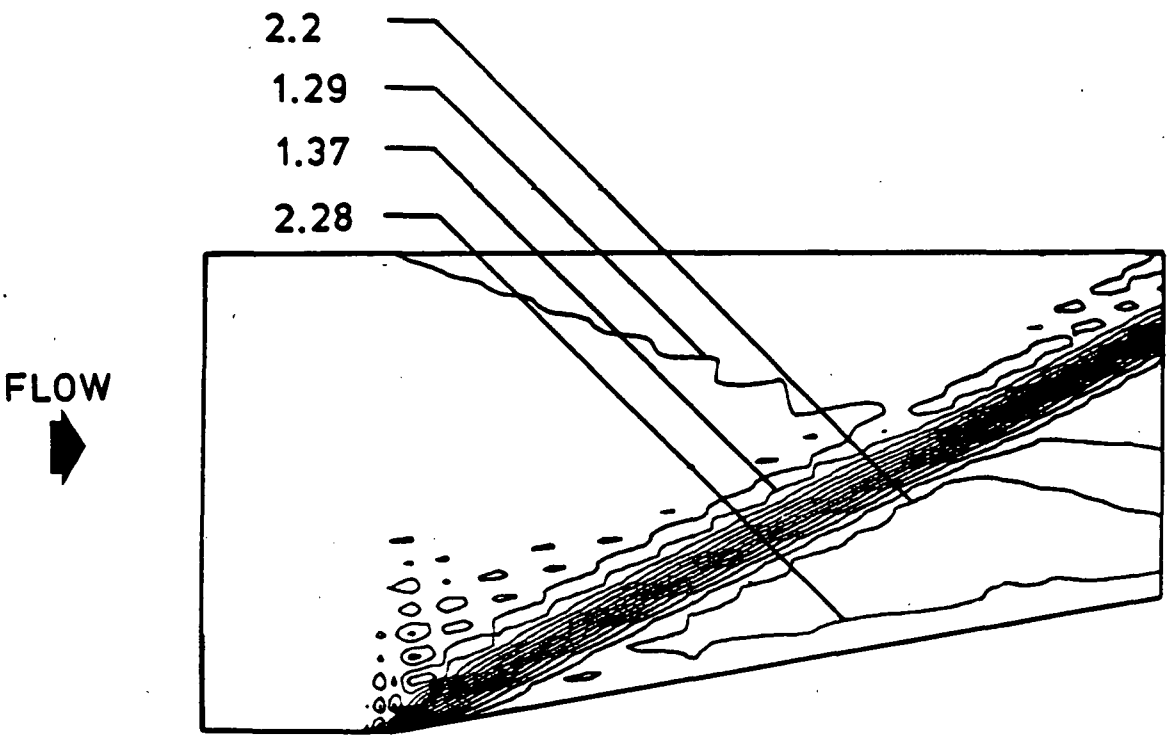


Fig. 4.2 Density contours for compression corner using Lapidus dissipation.

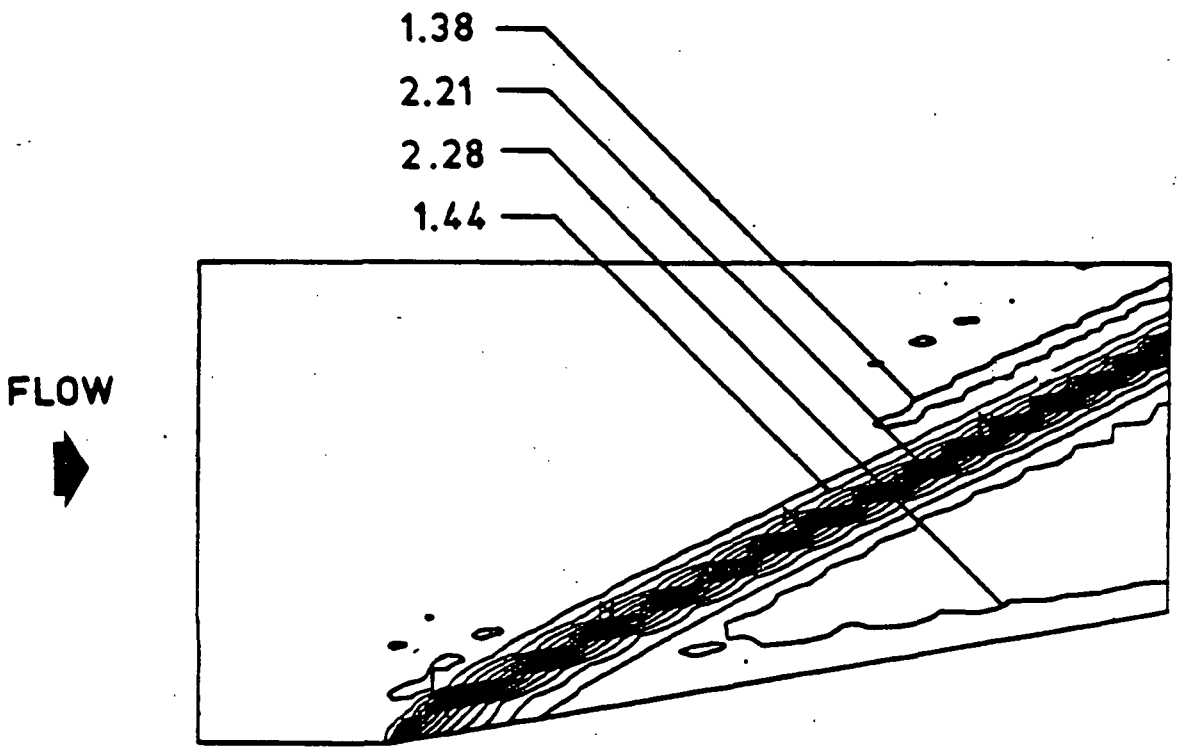


Fig. 4.3 Density contours for compression corner using MacCormack-Baldwin dissipation.

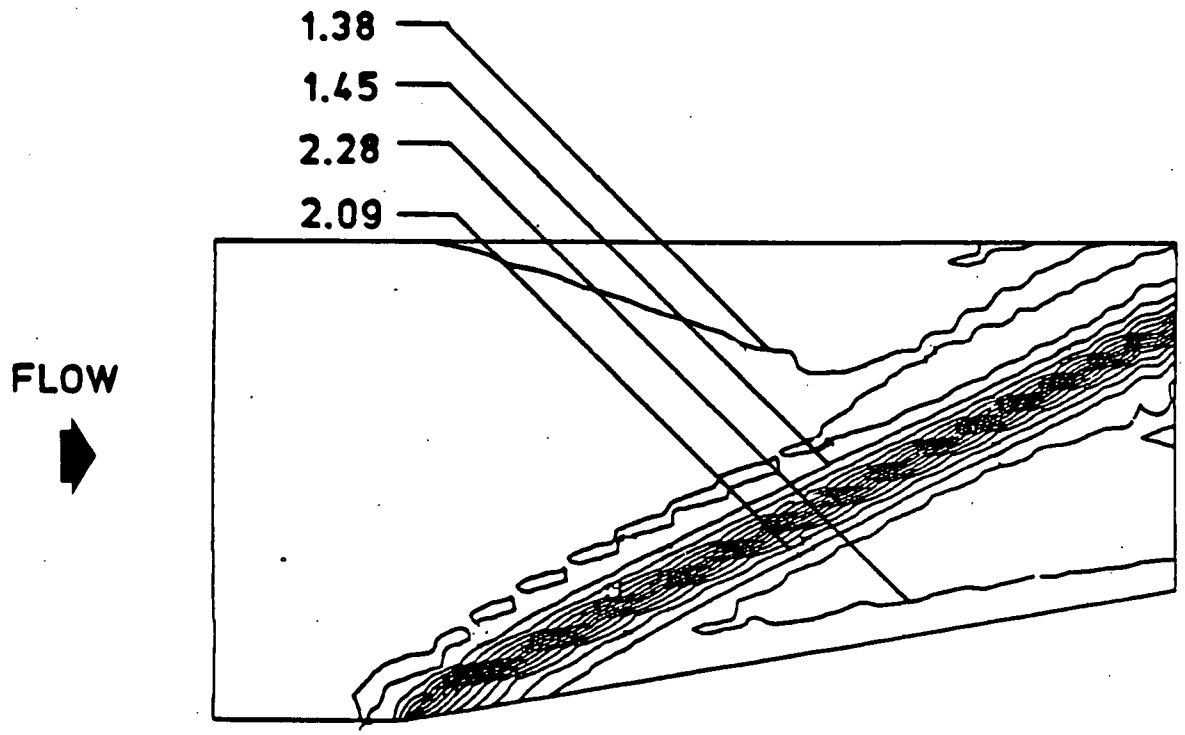


Fig. 4.4 Density contours for compression corner using Jameson dissipation model.

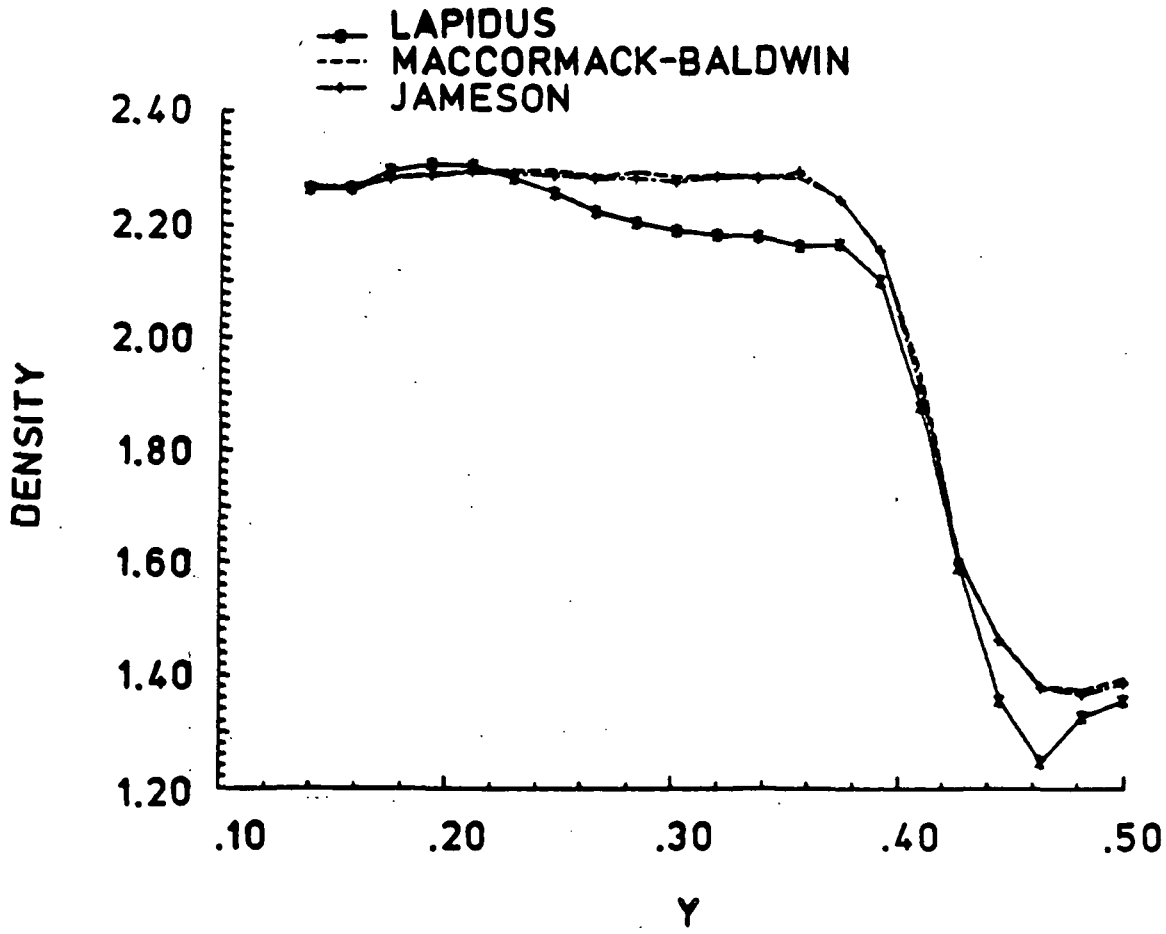


Fig. 4.5 Comparative density distribution at the outflow of compression corner for dissipation models.

The results obtained for this model problem indicate that the Lapidus model produces spurious oscillations at the shock and in the freestream. The results obtained using the MacCormack-Baldwin and Jameson operators are virtually identical. The Jameson model reduces freestream oscillations and improves convergence rates. The disadvantages of the Jameson model include the need for the expensive second and third derivative evaluations of the conservation variables. The scheme introduces two "tuning" parameters which may indicate the need for parametric studies to find the "right" values for the two constants for each problem of interest. The MacCormack-Baldwin dissipation model shows the least oscillations in the freestream, and the accuracy of the procedure is uniformly good. The results presented in the remainder of this chapter for the Taylor-Galerkin formulation were obtained using this dissipation model.

The use of triangular elements instead of "quad" elements for the Taylor-Galerkin formulation using dissipation a la Lapidus has shown a reduction in freestream oscillations [4]. For the triangular elements, the use of the MacCormack-Baldwin dissipation model has shown no significant advantages over the Lapidus model [42]. The contours of Figs. 4.2 and 4.3 suggest the Taylor-Galerkin formulation with quad elements is sensitive to the artificial dissipation model used. For quad elements significant improvements in solution quality are obtained using the MacCormack-Baldwin dissipation model.

4.2 Local Time-stepping Scheme

Many problems of concern to the aerodynamicist are steady state problems wherein the transient behavior is of little interest. The use

of time-steps that depend on the local flow conditions and geometry can help accelerate convergence. The allowable time-step varies throughout the mesh and is locally constrained by the CFL condition [43]. The CFL condition relates the propagation of information within the mesh to the grid spacing of the mesh. In two space dimensions the CFL condition limits the local time-step to $(\Delta t)_{CFL}$ given by,

$$(\Delta t)_{CFL} = \left[\left| \frac{u_i}{\Delta x_i} \right| + c \left(\frac{1}{\Delta x_i} \quad \frac{1}{\Delta x_i} \right)^{1/2} \right]^{-1} \tag{4.1}$$

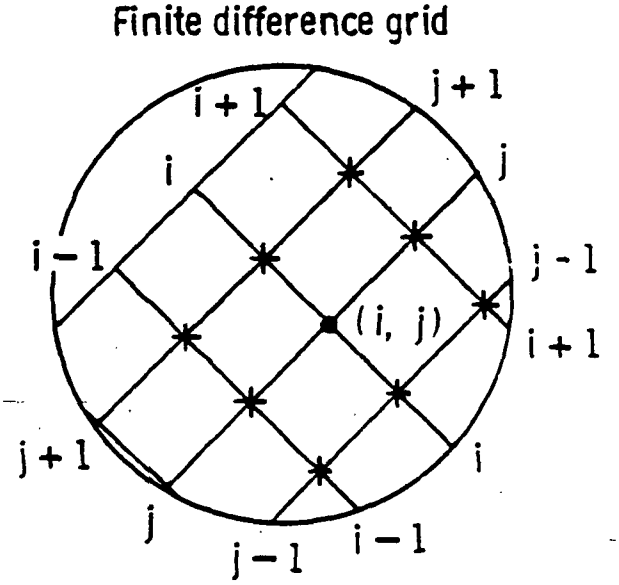
where u_i are the velocity components in the coordinate directions, Δx_i the grid spacing in the coordinate directions and c is the local speed of sound.

Finite difference methods use i-j grids where the geometry data is completely structured, Fig 4.6a. For finite elements the orientation of an element is random and the definition of Δx_i is not straightforward. For finite element meshes a better approach is to compute $\Delta\xi$ and $\Delta\eta$ as defined in Fig. 4.6b, where ξ and η are the local directions that depend on the orientation of the element. The CFL condition can then be written for a typical finite element as,

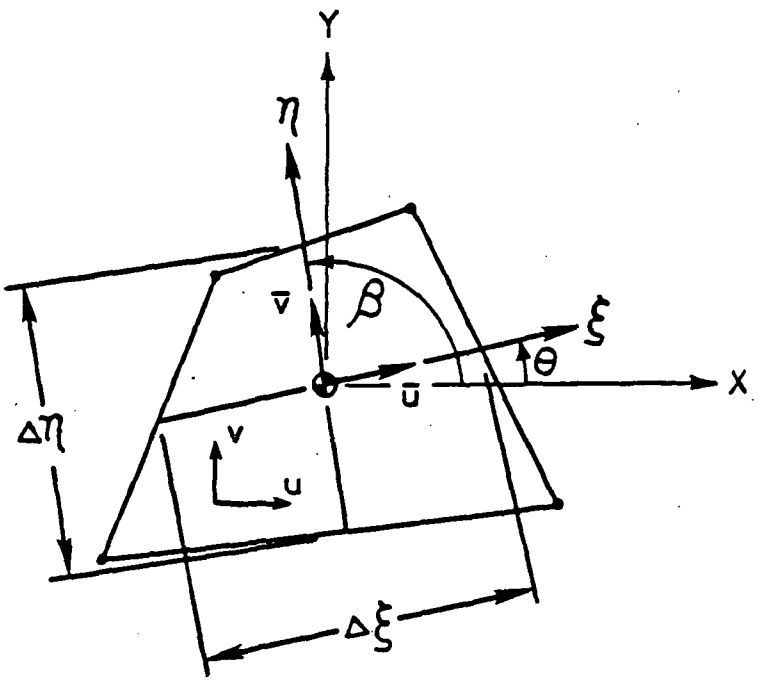
$$(\Delta t)_{CFL} = \left[\frac{\bar{u}_1}{\Delta\xi} + \frac{\bar{u}_2}{\Delta\eta} + c\sqrt{\frac{1}{(\Delta\xi)^2} + \frac{1}{(\Delta\eta)^2}} \right]^{-1} \tag{4.2}$$

where $\bar{u}_i$ are the local velocity components. The velocities $\bar{u}_i$ can be obtained by a transformation of the velocity components in the coordinate directions given by,

$$\begin{pmatrix} \bar{u}_1 \\ \bar{u}_2 \end{pmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ \cos\beta & \sin\beta \end{bmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \tag{4.3}$$



(a) Typical finite difference grid



(b) Typical quad element

Fig. 4.6 Definition of grid spacings for finite difference and finite element meshes.

where θ and β are the inclination angles of the local directions with the global coordinate axes.

The allowable time-step for an element is calculated from the relation,

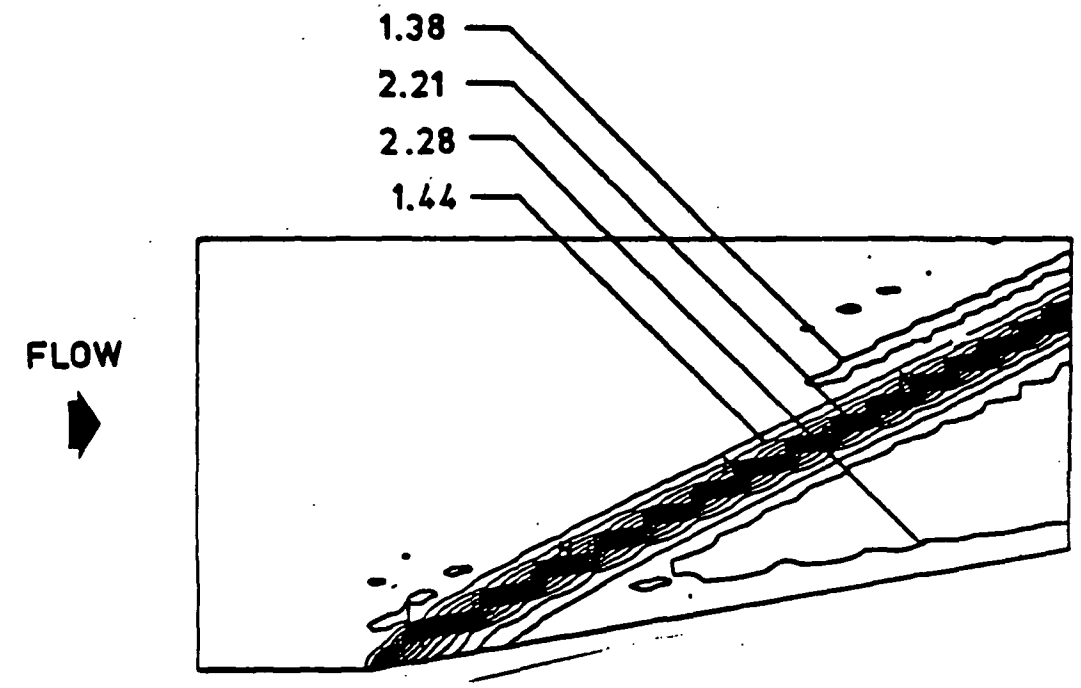
$$(\Delta t)_{ele} = \sigma \, (\Delta t)_{CFL} \tag{4.4}$$

where σ is a safety parameter that depends on the algorithm of choice. Finite element algorithms typically need local time-steps defined at the nodes in addition to those defined for the elements. The local time-step at a node is calculated based on the minimum element time-step of all the elements surrounding that node. For a typical node i, the local time-step is calculated from,

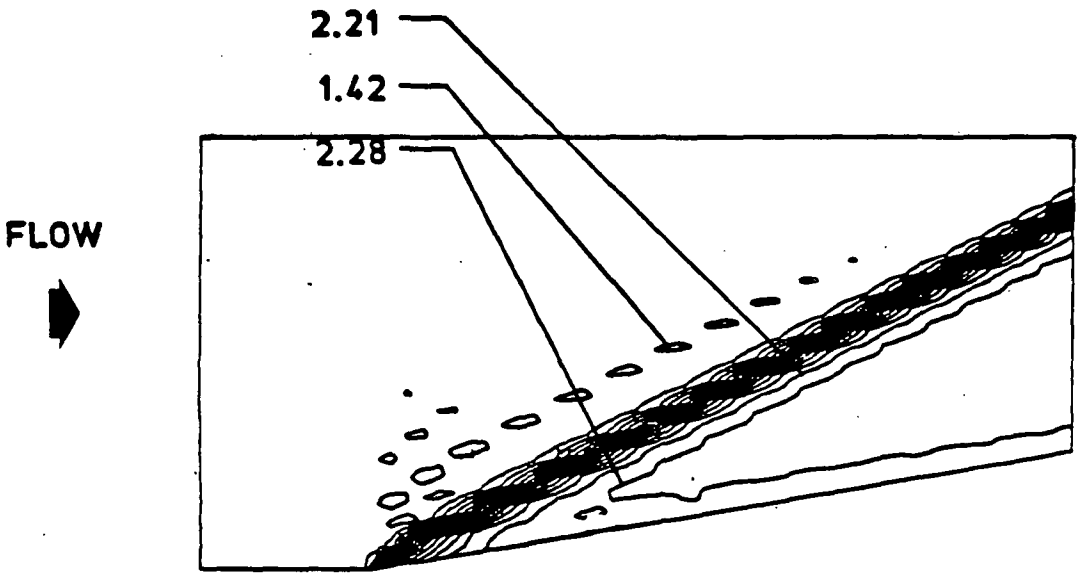
$$(\Delta t)_{node} = \min_{j = 1, \, nel} (\Delta t)^{j}_{ele} \tag{4.5}$$

where nel is the number of elements surrounding node i.

The benefits of using this time-stepping procedure are demonstrated by predicting the Mach 3 flow over the compression corner of Fig. 4.1. The artificial dissipation model used is the MacCormack-Baldwin model with a dissipation constant $\nu = 1$. Figure 4.7 contrasts the density contours for the Taylor-Galerkin procedure with global and local time-steps. The use of local time-steps is seen to reduce pre-shock oscillations at the top right corner of the flowfield. Figure 4.8 plots the density distribution at the outflow and indicates a sharper shock with the local time-stepping scheme with the shock being captured within 5 nodes instead of within 7 nodes. The use of local time-steps results in faster convergence rates as indicated by Fig. 4.9. The local time-stepping causes the L_2 norm of the density



(a) Density contours using global time-steps



(b) Density contours using local time-steps

Fig. 4.7 Comparison of the density contours for compression corner using local and global time-stepping schemes.

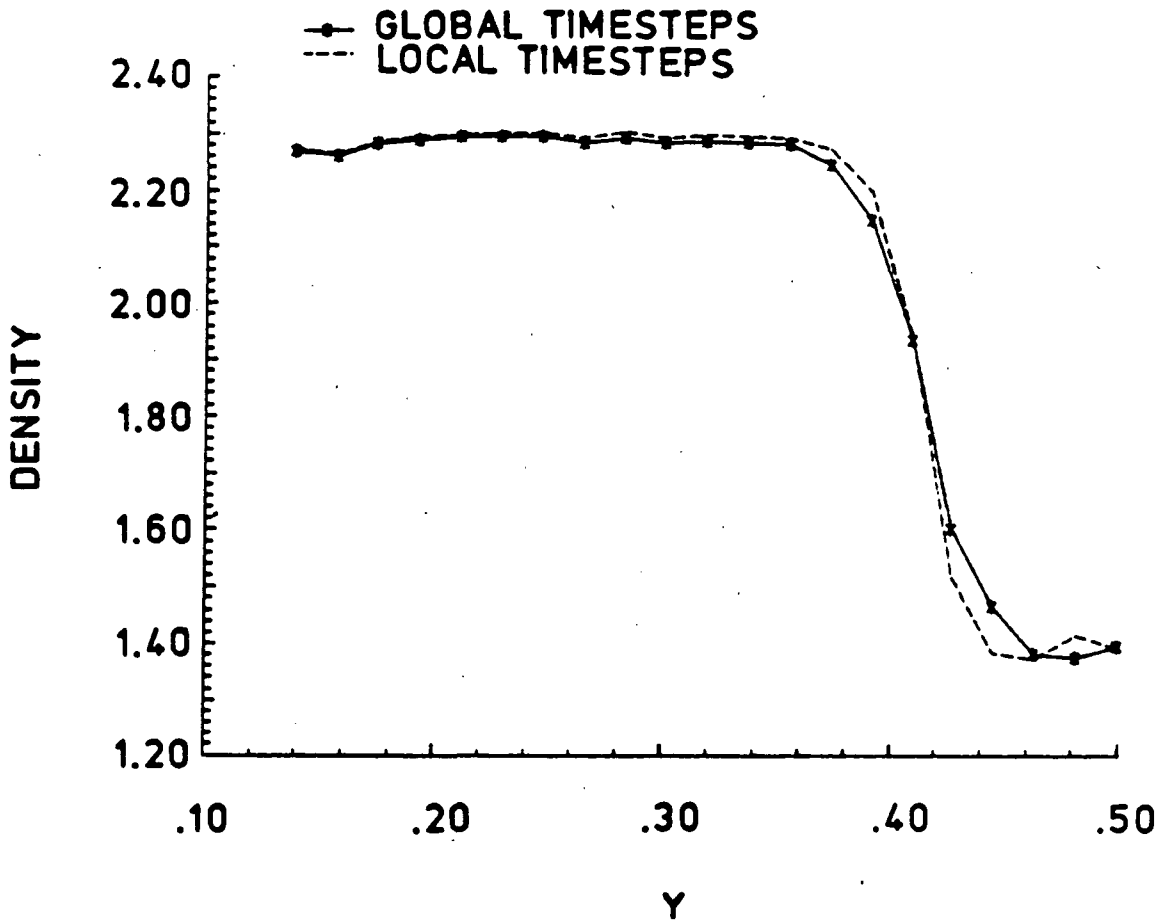


Fig. 4.8 Comparative density distributions at the outflow of compression corner for time-stepping schemes.

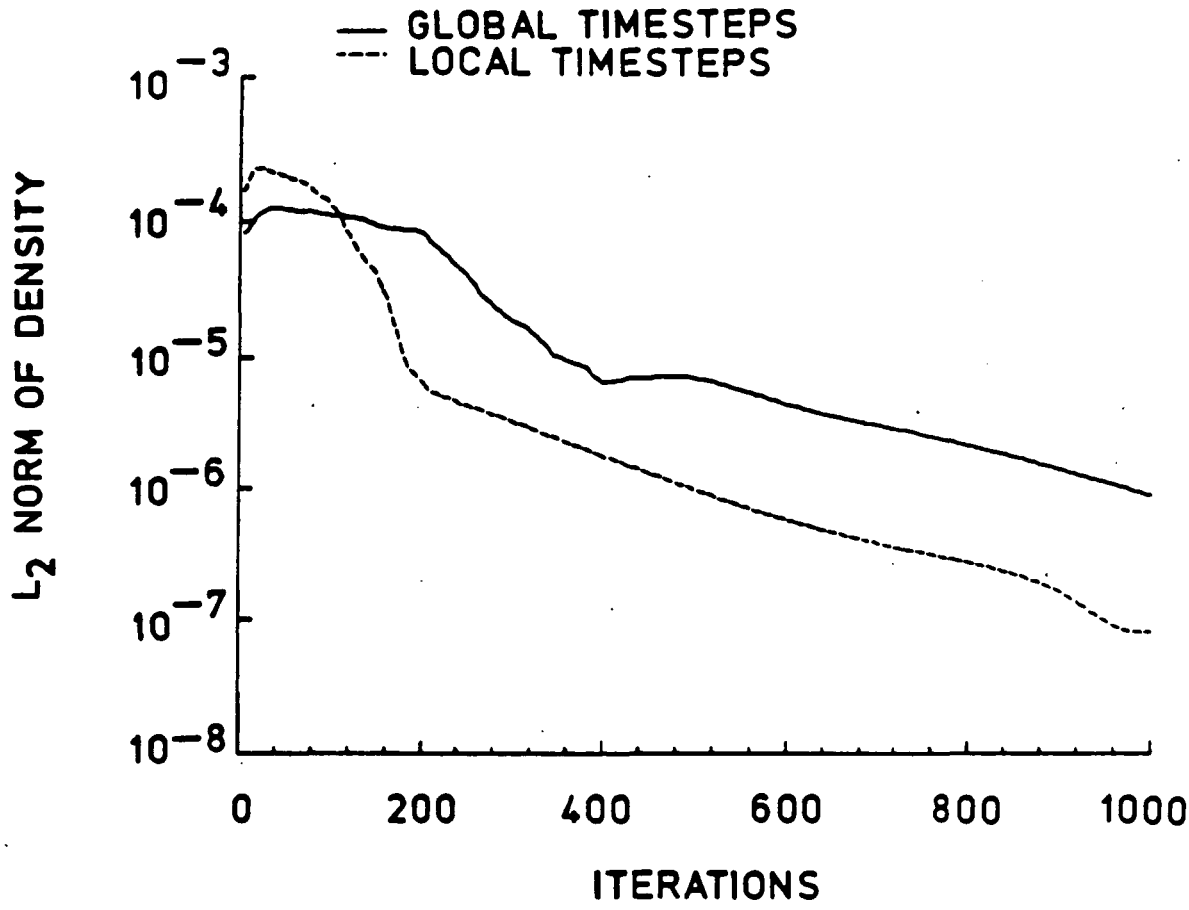


Fig. 4.9 Comparative convergence rates for compression corner using local and global time-stepping schemes.

changes to drop over two orders of magnitude within 200 iterations, compared to the 400 iterations needed for the global time-stepping procedure.

4.3 Evaluation of Inviscid Formulations

The Taylor-Galerkin formulation implemented with the MacCormack-Baldwin dissipation model and the local time-stepping procedure detailed in the previous section is compared with the Petrov-Galerkin formulation for a variety of problems. The Petrov-Galerkin formulation is also implemented with the local time-stepping scheme to provide a better basis for comparison. The evaluation of the finite element formulations is based on criteria which include solution accuracy, shock resolution, spurious oscillation control, computational speed, and storage requirements.

The problems that are used for the evaluation consist of: (1) Mach 3 flow over a compression corner, (2) Mach 6 expansion over a sharp corner, (3) interaction of a scramjet exhaust with the freestream, and (4) Mach 6.57 flow over a blunted leading edge.

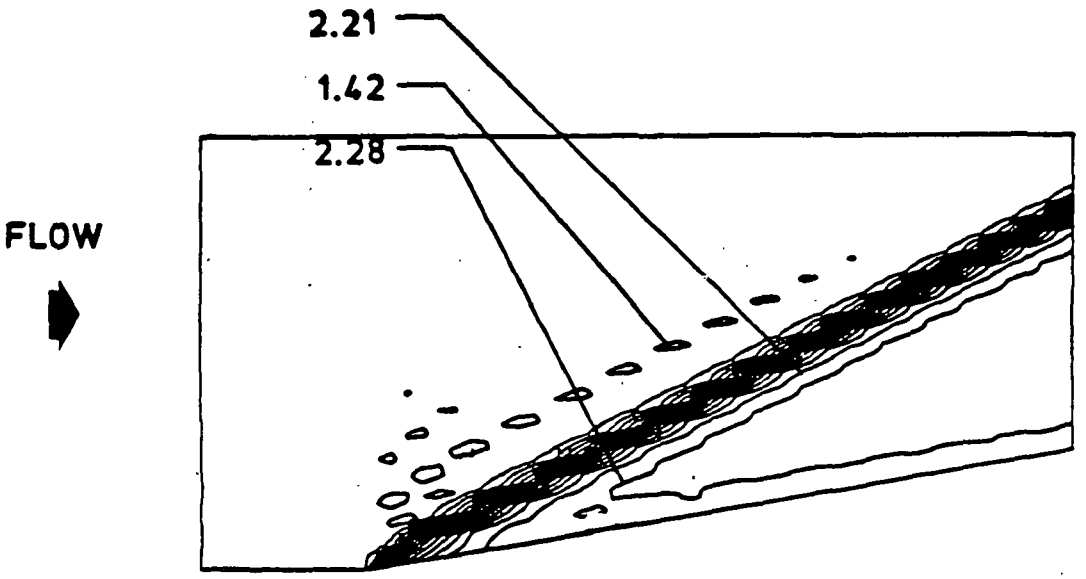
The compression corner and the Prandtl-Meyer expansion were chosen due to the availability of exact solutions. In addition to illustrating basic features of compressible flows such as shocks and expansions, these two problem helped validate the computer programs. The interaction of a scramjet exhaust with the freestream and the Mach 6.57 flow over the blunted body are chosen because of the availability of solutions by other numerical methods such as finite volume or finite difference methods. These problems are also typical of the flow situations encountered in realistic problems.

The Taylor-Galerkin and the Petrov-Galerkin formulations were used to predict the flow behavior for the problems listed above. The Taylor-Galerkin formulation with the MacCormack-Baldwin dissipation model uses a dissipation constant of $\nu = 1$ for the first three problems. Results obtained for the hypersonic flow over the blunt leading edge indicated the need for a dissipation constant of $\nu = 2$ to suppress shock related oscillations.

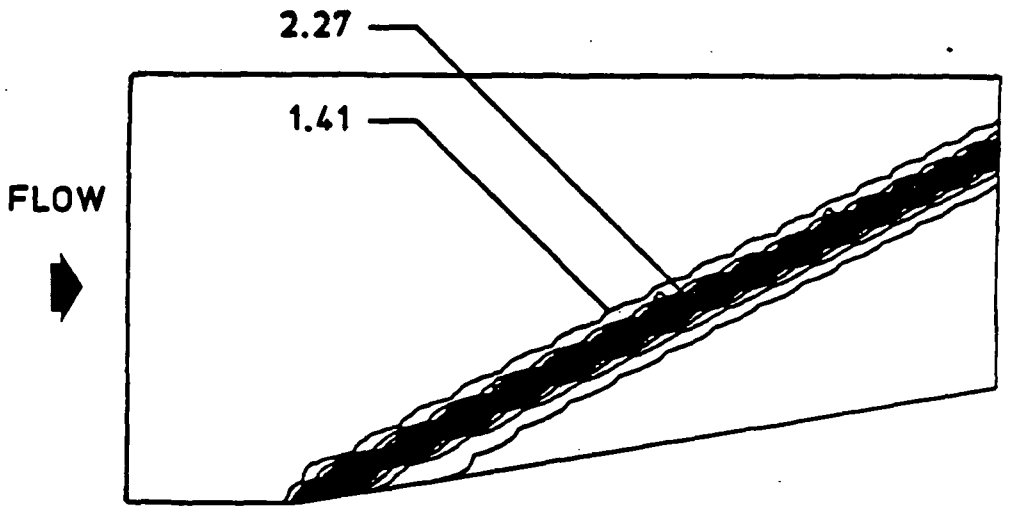
4.3.1 Compression Corner

The density contours for the compression corner using the Taylor-Galerkin and Petrov-Galerkin formulations appear in Fig. 4.10. The results obtained for the Taylor-Galerkin formulation indicate the smeared shock along the wall and other flow details that have been discussed in section 4.2. Results obtained using the Petrov-Galerkin formulation indicate the absence of spurious oscillations. The shock obtained is very crisp as indicated by the contour levels running close together, and the density at the wall shows no oscillations.

The distribution of density along the wall and at the outflow plane for the two methods are compared in Fig. 4.11. Both methods show good shock capturing properties with the shock defined within five nodes. The Taylor-Galerkin algorithm exhibits a little undershoot at the corner and spurious oscillations occur along the wall. The presence of these oscillations is seen clearly by the distributions of Fig 4.11b. The results obtained by the Petrov-Galerkin formulation show no such oscillations, and the density at the wall compares exactly with analytical solutions.

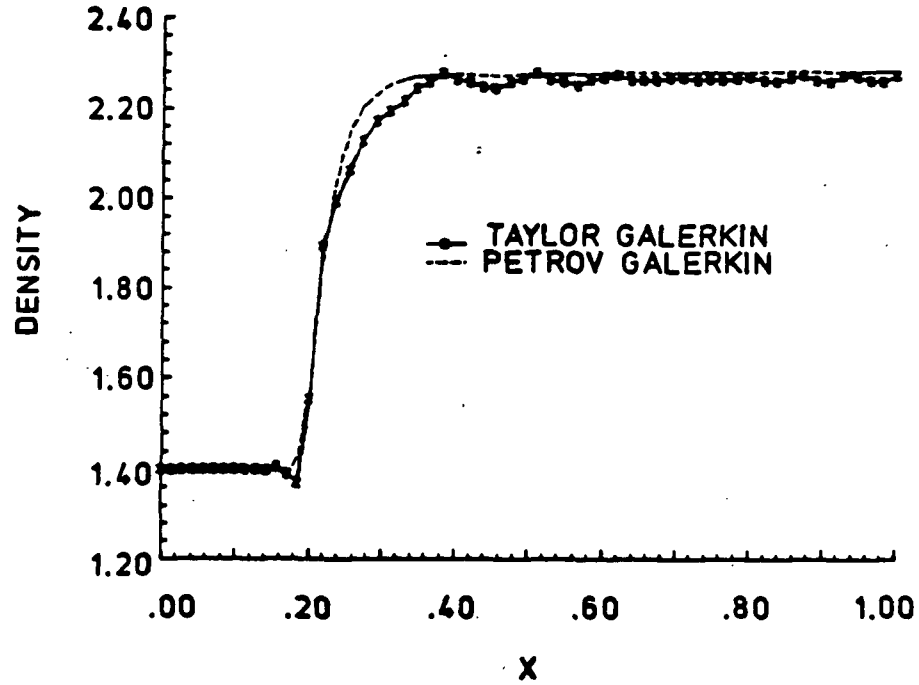


(a) Taylor-Galerkin contours

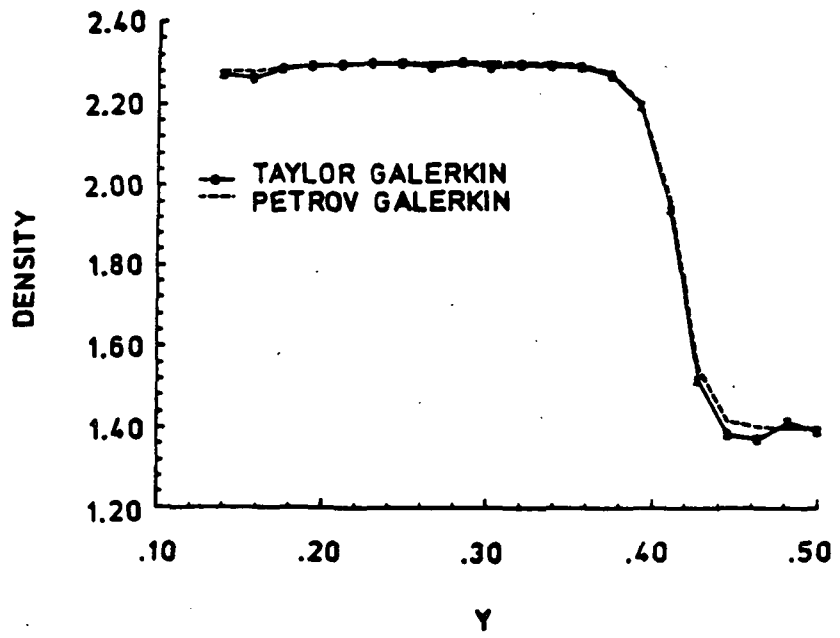


(b) Petrov-Galerkin contours

Fig. 4.10 Comparison of the density contours for the compression corner.



(a) Density distributions along the wall



(b) Density distributions at the outflow

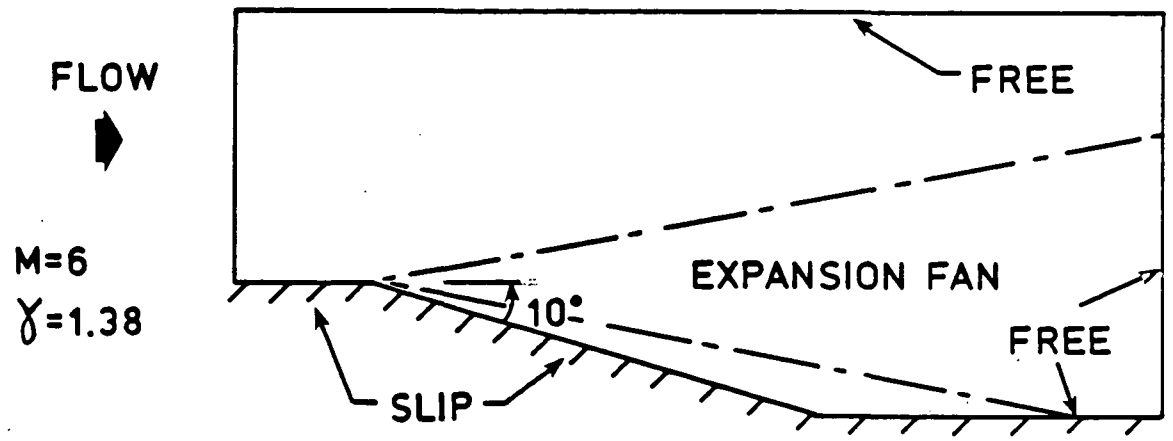
Fig. 4.11 Comparative density distributions along the wall and at the outflow for compression corner.

4.3.2 Prandtl-Meyer Expansion

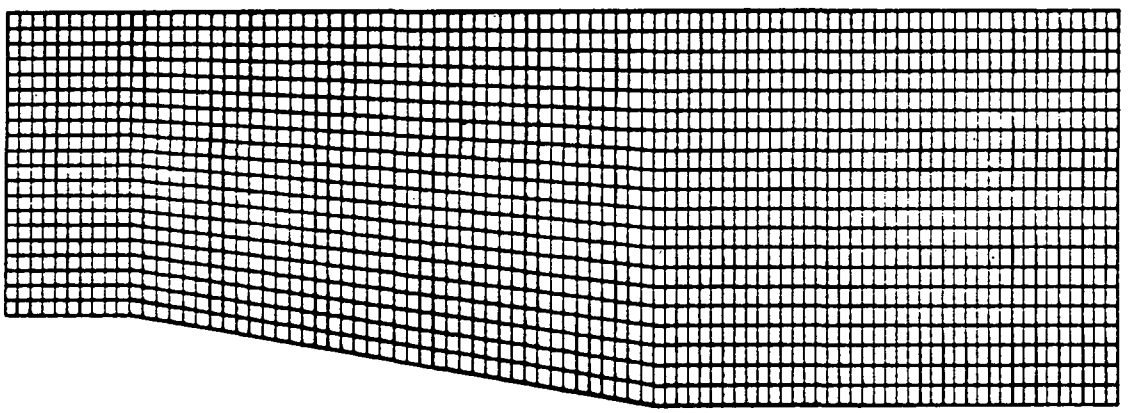
The capability of the finite element formulations for detailing expansion waves is illustrated by predicting the features of a Mach 6 flow over a 10° corner. The flow parameters after the expansion can be obtained from isentropic relations. The flow configuration and the boundary conditions for the region are shown in Fig. 4.12a. A finite element mesh containing 1800 quad elements and 1920 nodes is used to predict the effects of expanding the Mach 6 flow through 10°.

The density contours for the Taylor-Galerkin and Petrov-Galerkin formulations appear in Fig. 4.13. The contours for the Taylor-Galerkin indicate the presence of a few oscillations at the root of the expansion fan and an overshoot of about 5% in the freestream. The density contours for the Petrov-Galerkin formulation show very little oscillations. The contour levels indicate a smooth transition through the expansion fan. The root of the expansion fan is smeared along the wall, but the smearing is not as severe as that obtained using the Taylor-Galerkin formulation.

The distributions of density at the outflow plane and along the wall for the two methods appear in Fig. 4.14. The distributions at the outflow plane indicate the smooth transition of values from the freestream to the values at the tail of the expansion fan. The presence of the slight overshoot mentioned earlier for the Taylor-Galerkin formulation is seen more clearly from this distribution plot. The density distributions along the wall indicate the presence of a kink at the corner for the Taylor-Galerkin. The results of the Petrov-Galerkin formulation indicate the presence of a few oscillations near the corner, but the expansion predicted at the wall is smoother.

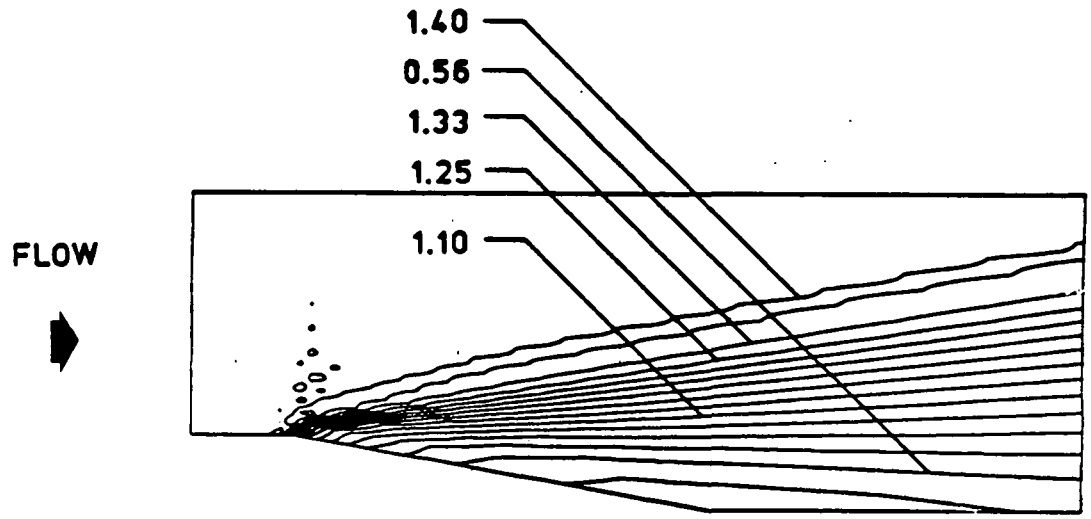


(a) Flow configuration

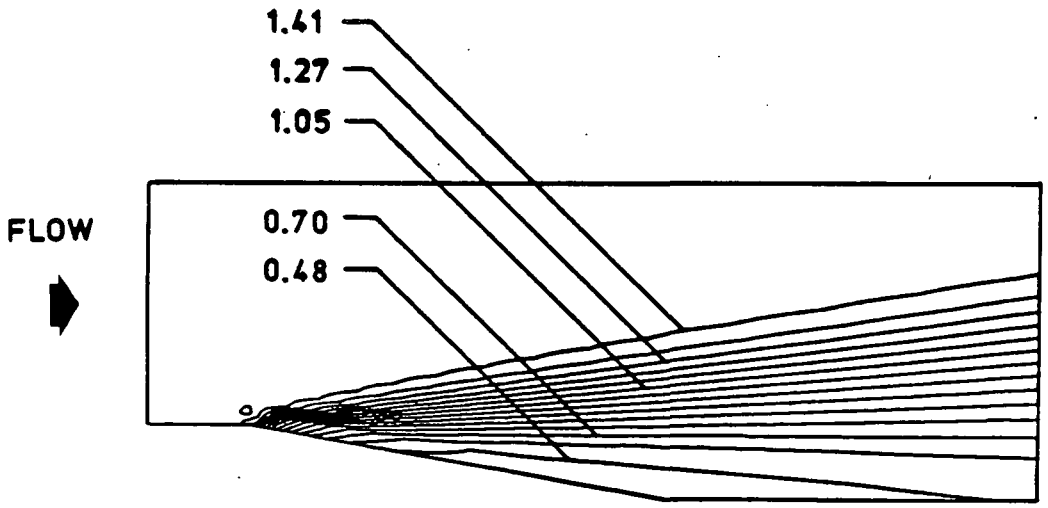


(b) Finite element mesh

Fig. 4.12 Flow configuration and finite element mesh for Prandtl-Meyer expansion.

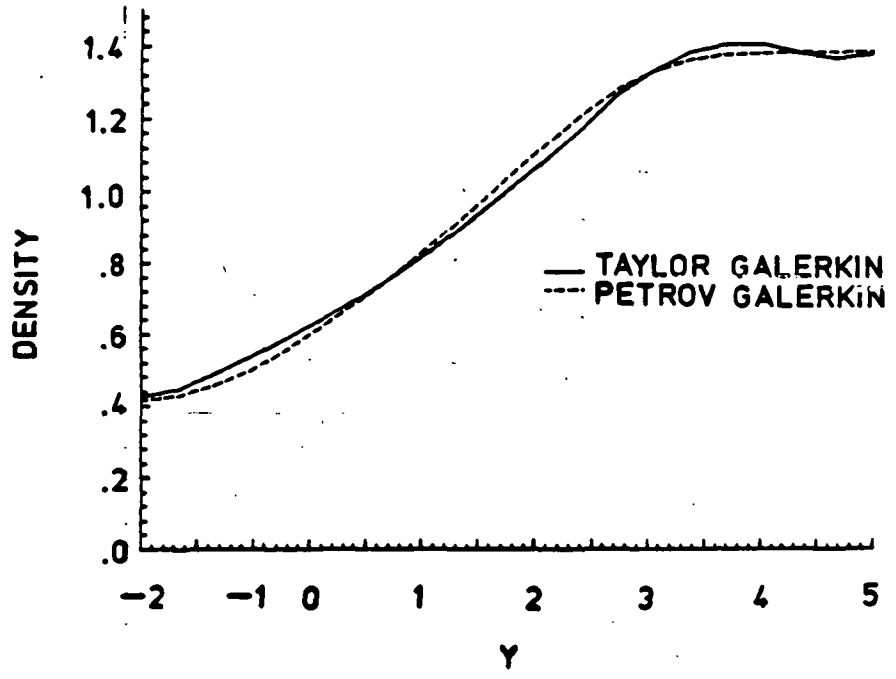


(a) Taylor-Galerkin contours

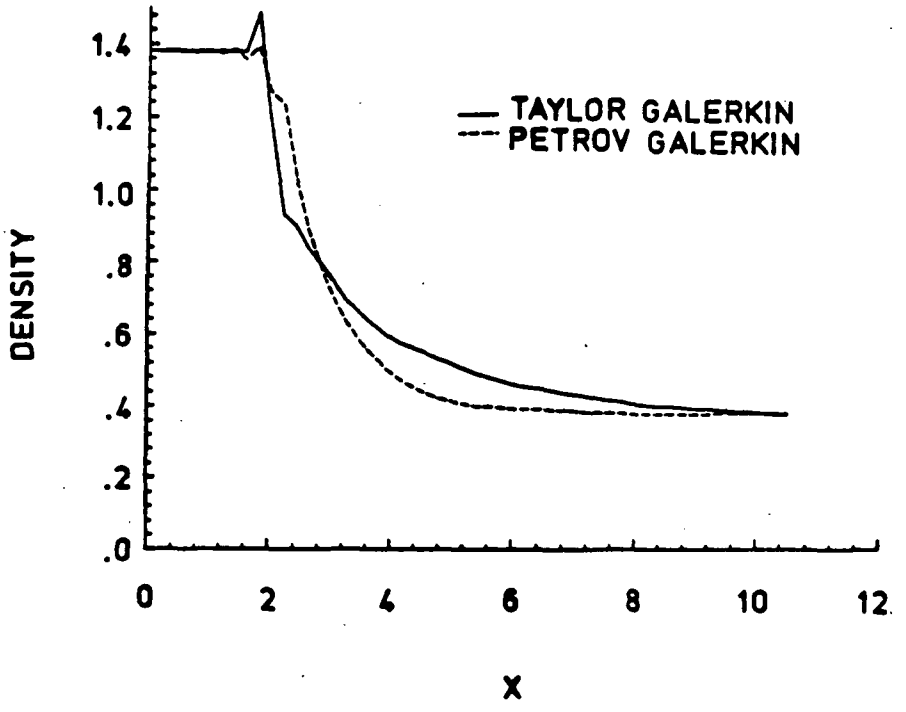


(b) Petrov-Galerkin contours

Fig. 4.13 Comparison of density contours for the Prandtl-Meyer expansion.



(a) Density distributions at the outflow



(b) Density distributions along the wall

Fig. 4.14 Comparative density distributions at the outflow and along the wall for the Prandtl-Meyer expansion.

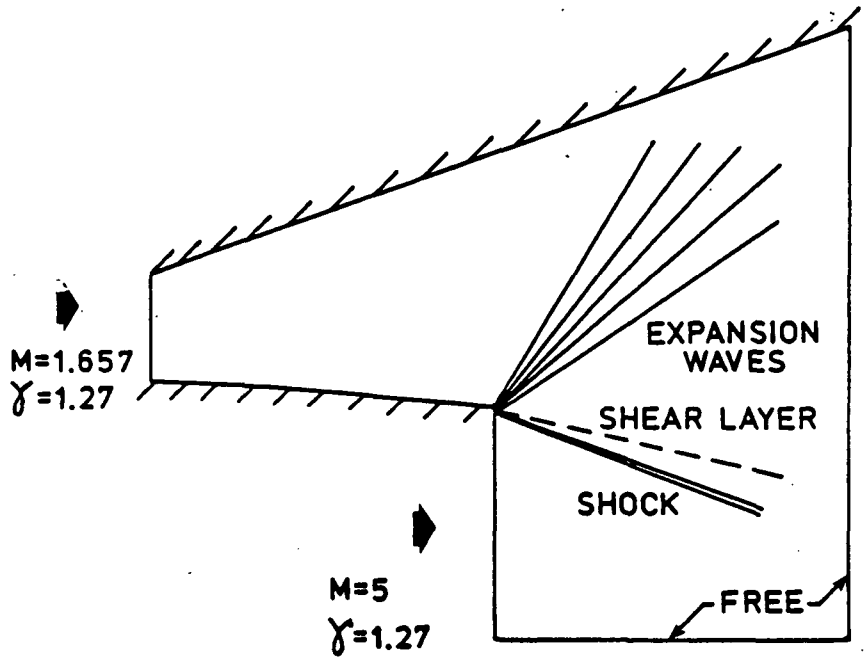
Adequate resolution of the root of the expansion fan is critical for obtaining accurate solutions throughout the flowfield. The distributions of Fig. 4.14 indicate the need for more refinement at the corner to capture essential details of the expansion.

4.3.3 Scramjet Exhaust Flow

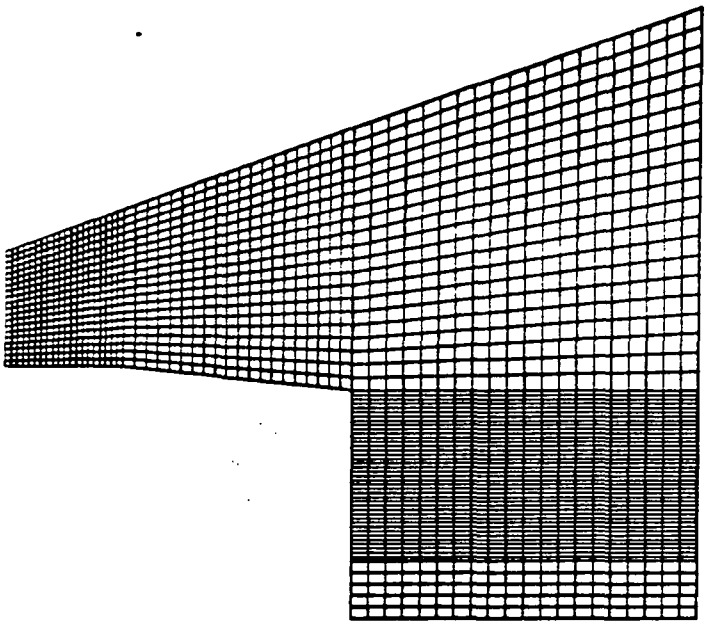
The scramjet has come under renewed scrutiny due to its potential applications in hypersonic research vehicles. The exhaust from the scramjet engine interacts with the freestream at the exit producing a shear layer and a shock; the predominant flow features that result due to this interaction are shown in Fig. 4.15a. The finite element mesh used to predict the flow feature appears in Fig. 4.15b. The mesh contains 2100 elements and 2226 nodes and is refined at regions where the gradients of the flow variables are expected to be large.

The density contours obtained for the finite element formulations appear in Fig 4.16. The expansion through the nozzle exhaust is indicated by the contour levels (I-O). The interaction of the expanded flow with the freestream at the bottom results in the shock and shear layer shown. The figures indicate the good shock capturing capabilities of both methods. The Taylor-Galerkin formulation shows a few localized oscillations at the corner where the flow from the nozzle exit interacts with that from the freestream. The shock location for the Petrov-Galerkin is seen to be sharper and closer to the shear layer. Details of the flowfield can be better shown by the distribution of quantities such as pressure and Mach number at the outflow.

Figure 4.17a compares the distribution of pressure at the outflow for both methods and the difference in the shock location is clearly

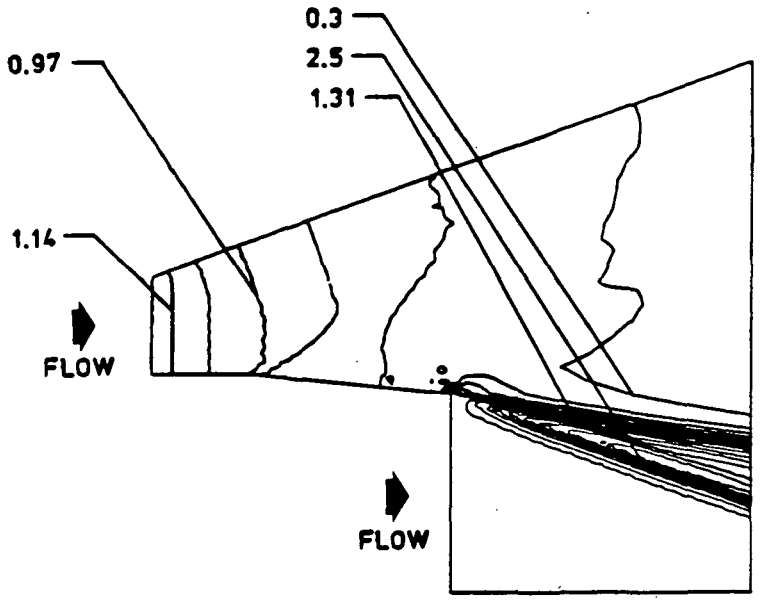


(a) Flow features

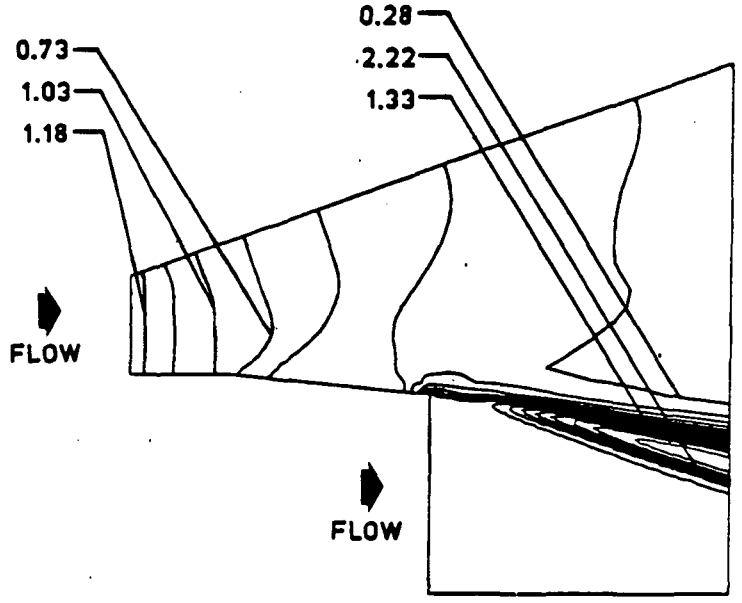


(b) Finite element mesh

Fig. 4.15 Flow configuration and finite element mesh for scramjet exhaust interaction.



(a) Taylor-Galerkin contours



(b) Petrov-Galerkin contours

Fig. 4.16 Comparison of density contours for the scramjet exhaust interaction.

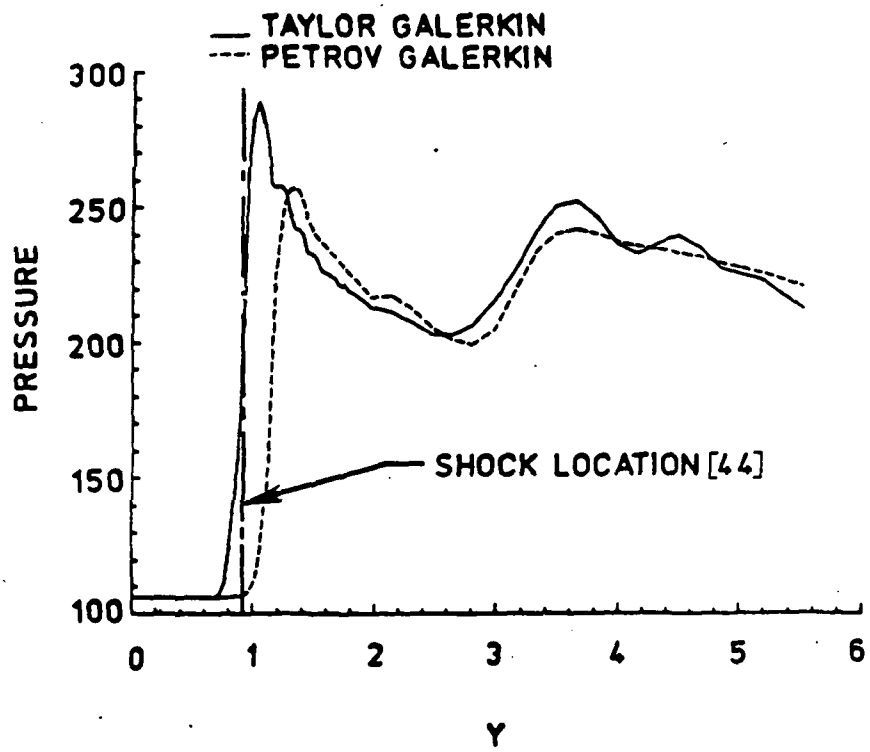
seen. The shock location predicted by the finite difference program SEAGULL [44] is also shown for comparison with the finite element results. The Taylor-Galerkin formulation is seen to be in excellent agreement with the finite difference code for the shock location while the Petrov-Galerkin formulation predicts a weaker shock shifted closer to the shear layer.

The distribution of Mach numbers along the outflow plane is plotted in Fig. 4.17b. The sharp drop in the Mach number at around y = 1.4 indicates the location of the shear layer. Flow exhausting from the scramjet interacts with the freestream along the line defining the shear layer.

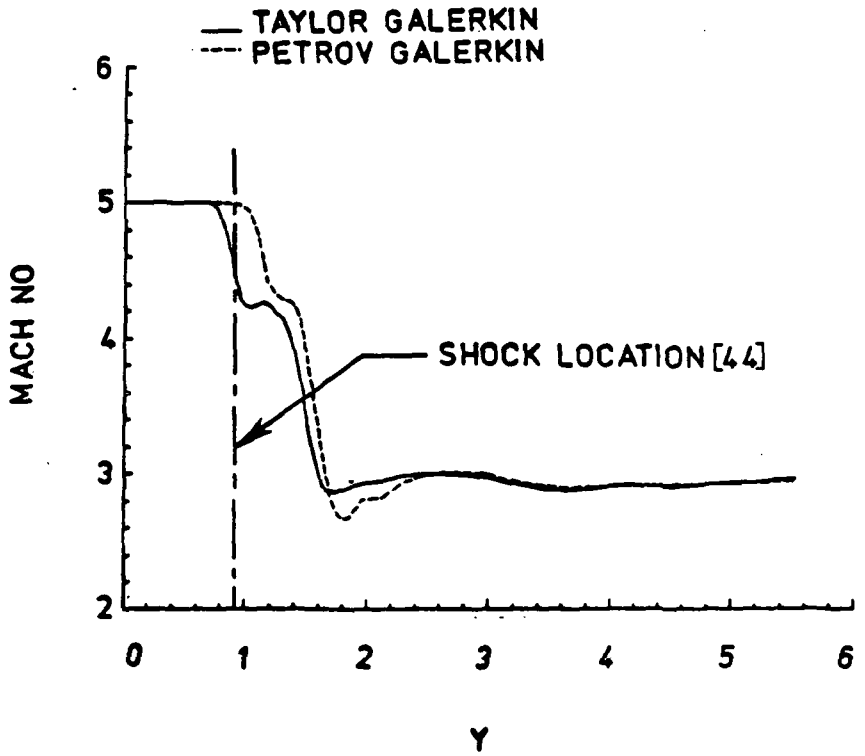
4.3.4 Blunt Leading Edge

The Aerothermal Loads Branch at the NASA Langley Research Center uses the 8' High Temperature Tunnel to test a variety of structural configurations. The panel holder used in testing has a blunt leading edge, and the finite element formulations are used to simulate the flow over the blunt section. The geometric configuration and the finite element mesh used for the analysis is shown in Fig 4.18.

Figure 4.19 shows the density contours obtained for the Taylor-Galerkin and Petrov-Galerkin formulations. The location of the shock in front of the leading edge and the shock location at the outflow are seen to be radically different for the two methods. The density contours for the Taylor-Galerkin formulation are smooth, and no post-shock oscillations are visible. The shock standoff distance is seen to be considerably smaller for the Petrov-Galerkin algorithm. The density levels at the body, especially at the outflow, is lower than that



(a) Pressure distributions at the outflow



(b) Mach number distributions at the outflow

Fig. 4.17 Comparative Pressure and Mach number distributions at the outflow for the scramjet exhaust interaction.

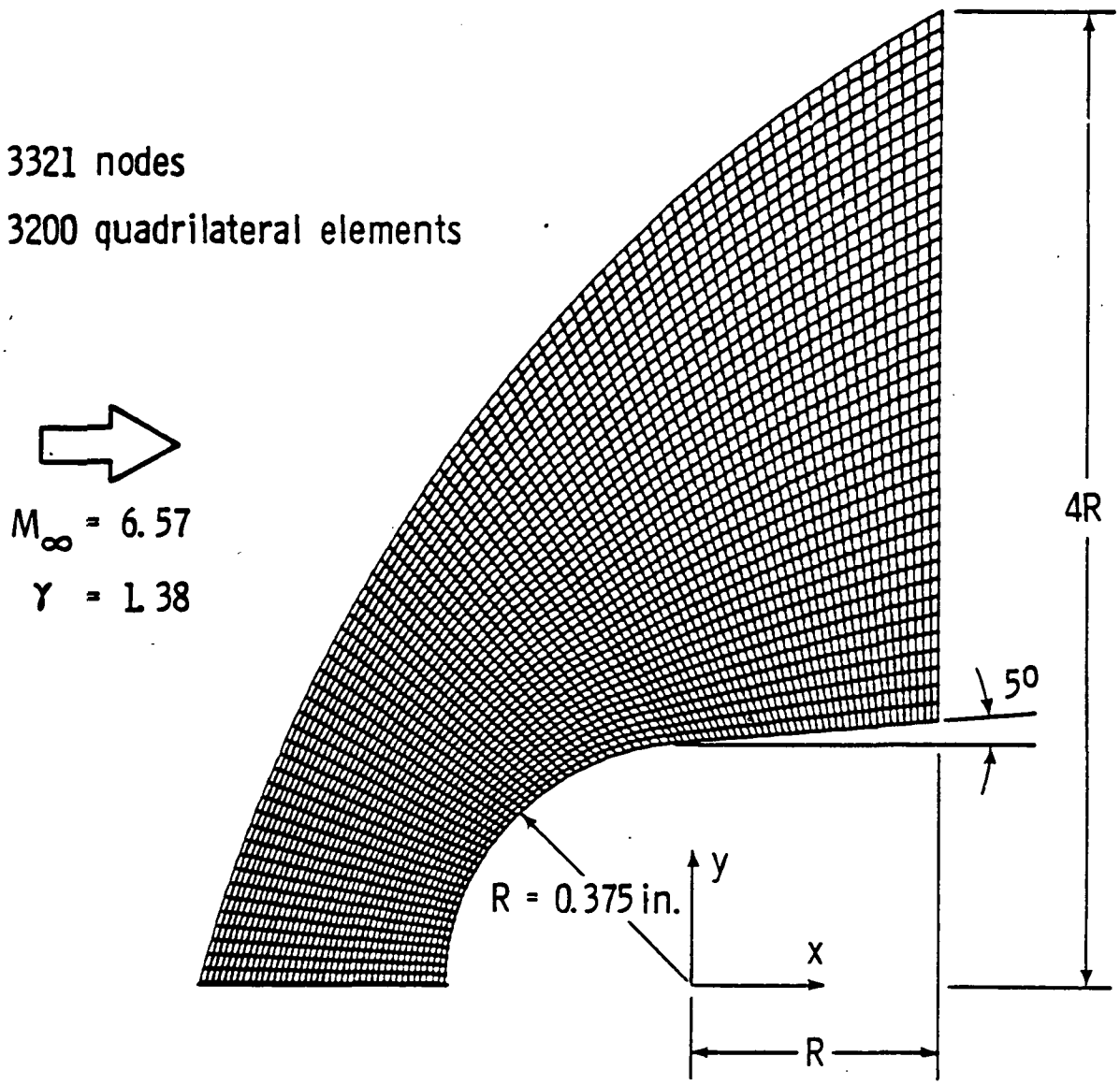
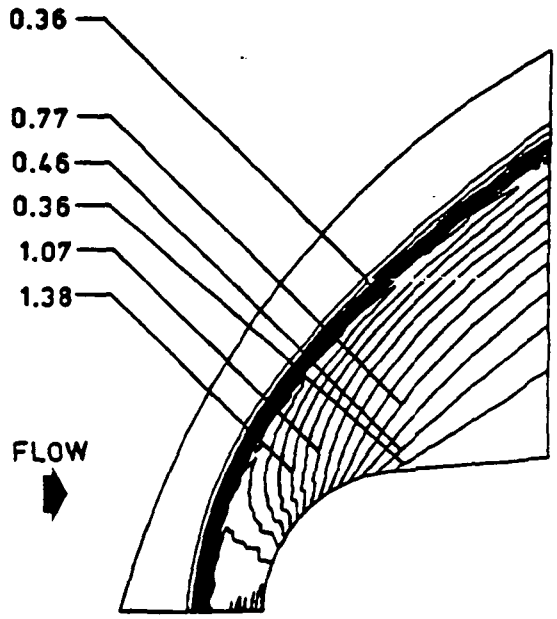
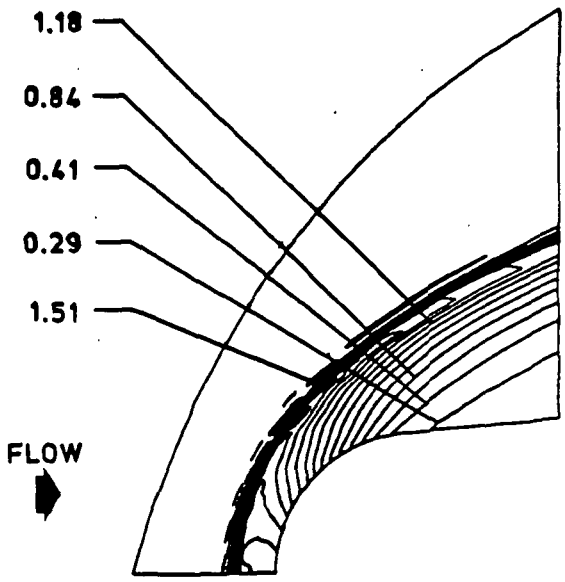


Fig. 4.18 Flow configuration and finite element mesh for Mach 6.57 flow over a blunt body.



(a) Taylor-Galerkin contours



(b) Petrov-Galerkin contours

Fig. 4.19 Comparison of density contours for blunt leading edge

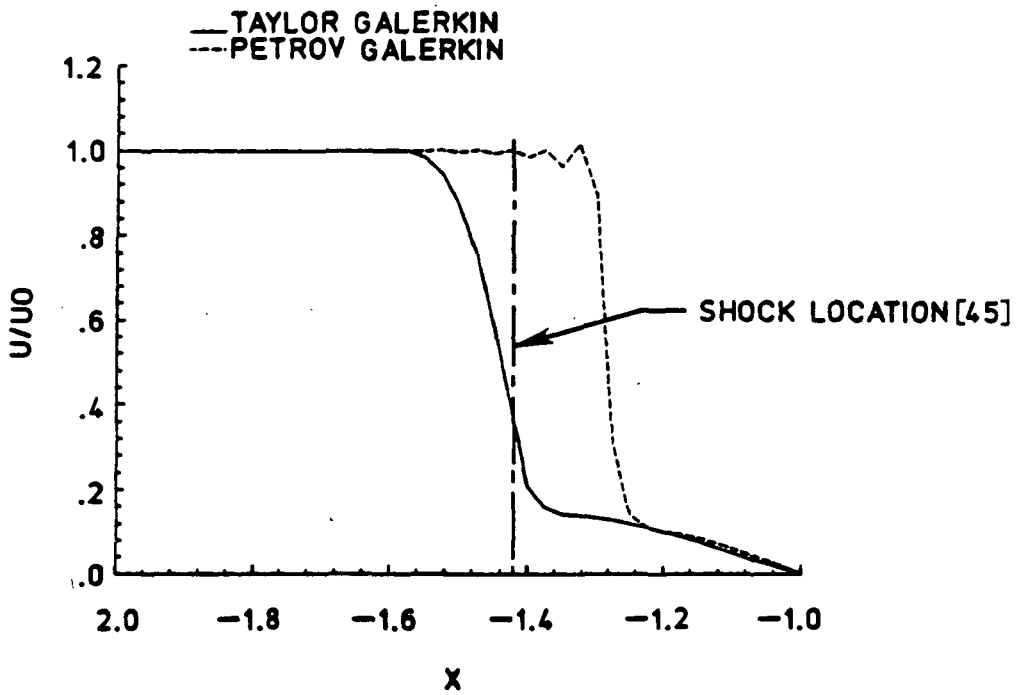
predicted by the Taylor-Galerkin formulation. The presence of post-shock oscillations for the Petrov-Galerkin formulation is also evident from the density contours.

The shock locations predicted by the two finite element formulations along the centerline are compared with the shock location given by the empirical relation of Billig [45] in Fig 4.20a. The Taylor-Galerkin formulation is seen to predict the shock location better than the Petrov-Galerkin. The comparison of the u-velocity component at the outflow for the two finite element formulations appears in Fig. 4.20b. The results obtained by the finite element methods are also compared with the finite volume results of Walters [46] for the same problem. The Taylor-Galerkin results show a few oscillations close to the surface of the body. The location of the shock predicted by the Taylor-Galerkin and the finite volume method are close, but the shock location of the Petrov-Galerkin formulation is clearly in error. The velocity at the body for the Petrov-Galerkin formulation is also higher which can be related to the low density levels at the wall.

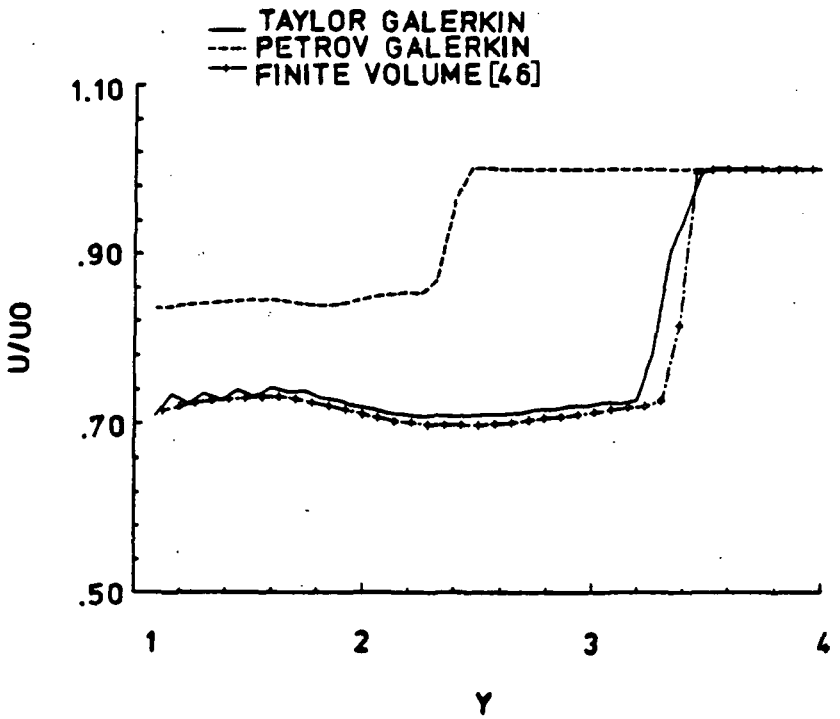
Recently a modification of the Petrov-Galerkin formulation was proposed [18] to make the method more "conservative." Better conservation of the flux quantities are obtained by integrating by parts the convective term in the weighted residual formulation. The weighted residual formulation is given by Eq (2.64) as,

$$\int_A N^T A_o V_{,t} \, dA = -\int_A W^T \tilde{A}_i V_{,i} \, dA \tag{4.6}$$

The use of Eq. (4.6) ensures conservation as long as the numerical integration procedure is of sufficient accuracy. The loss of



(a) Velocity distributions along centerline



(b) Velocity distributions at the outflow

Fig. 4.20 Comparative velocity distributions along the centerline and at the outflow for the blunt leading edge.

conservation is of the same order as the error in approximations using numerical quadrature. To circumvent the possibility of loss of conservation the weighted residual formulation can be recast by using an integration by parts procedure. The balance law for the Euler equations defines the transformed flux quantities $\tilde{F}_i$ as,

$$\tilde{F}_{i,i} = \tilde{A}_i \, V_{,i} \qquad (i \text{ not summed}) \tag{4.7}$$

Using Eq. (4.7) the advective terms can be written as,

$$\int_A W^T \tilde{F}_{i,i} \, dA = -\int_A W^T_{,i} \, \tilde{F}_i \, dA + \int_S W^T \tilde{F}_i n_i \, ds \tag{4.8}$$

The use of the integration by parts procedure ensures conservation of the advective flux especially when approximate integral evaluation procedures are used.

Researchers at Stanford University have used the Petrov-Galerkin formulation with this modification to predict the flow over the hypersonic blunt body for the flow parameters and the finite element mesh given in Fig. 4.18. The density contours obtained for the revised Petrov-Galerkin formulation appear in Fig. 4.21. The contours indicate a very well defined shock and the absence of oscillations throughout the flowfield. A comparison of the shock standoff distance at the centerline appears in Fig. 4.22 and compares very well with the prediction of Billig [45]. The results obtained from the Petrov-Galerkin formulation are also compared with an interpolation between the finite difference results for Mach 6 and Mach 8 flows [47].

The distribution of the u-velocity component at the outflow for the Petrov-Galerkin formulation is compared with the results of Walters [46] in Fig 4.23. The results indicate excellent agreement with the

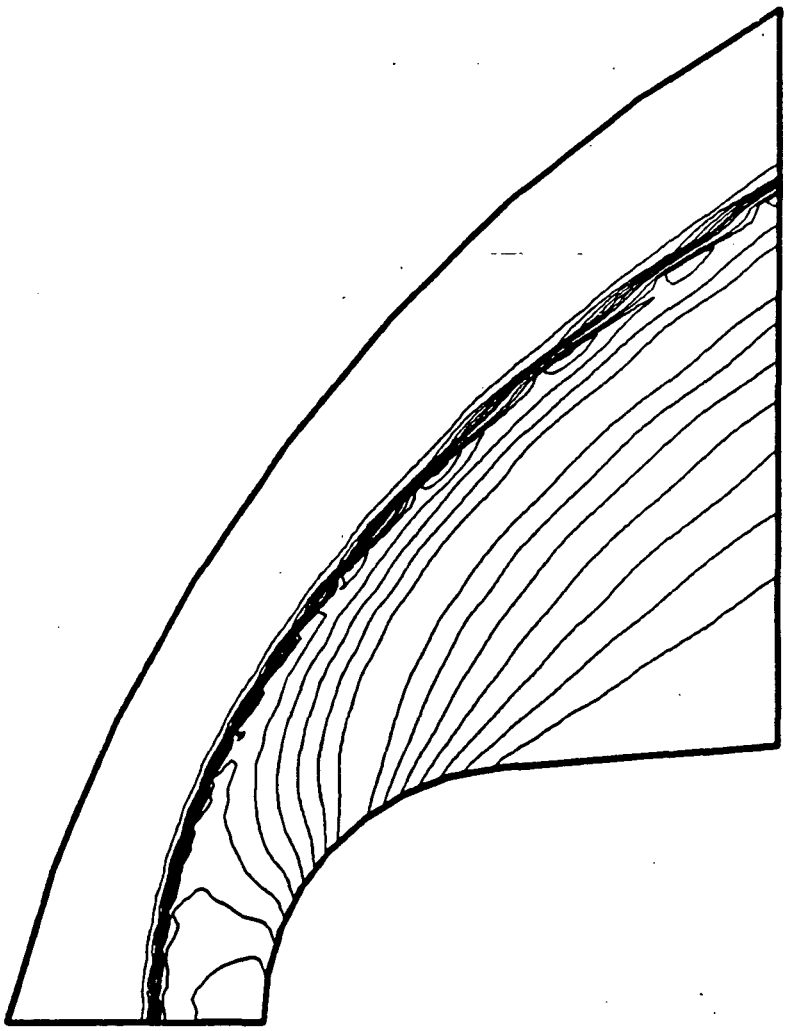


Fig. 4.21 Density contours for blunt leading edge using a modified Petrov-Galerkin formulation, Ref. [18].

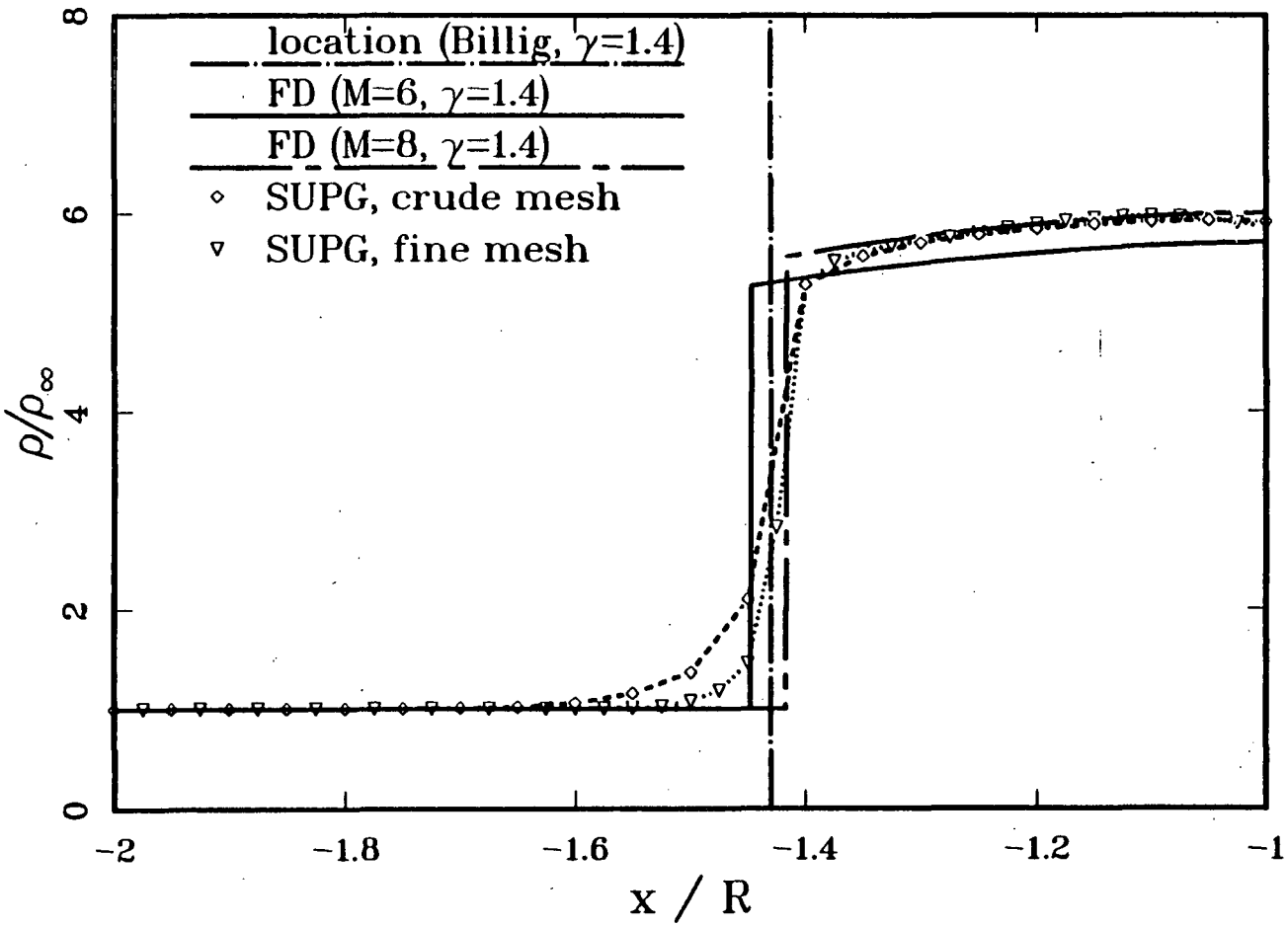


Fig. 4.22 Comparative density distributions at the centerline for the blunt leading edge using a modified Petrov-Galerkin formulation, Ref. [18].

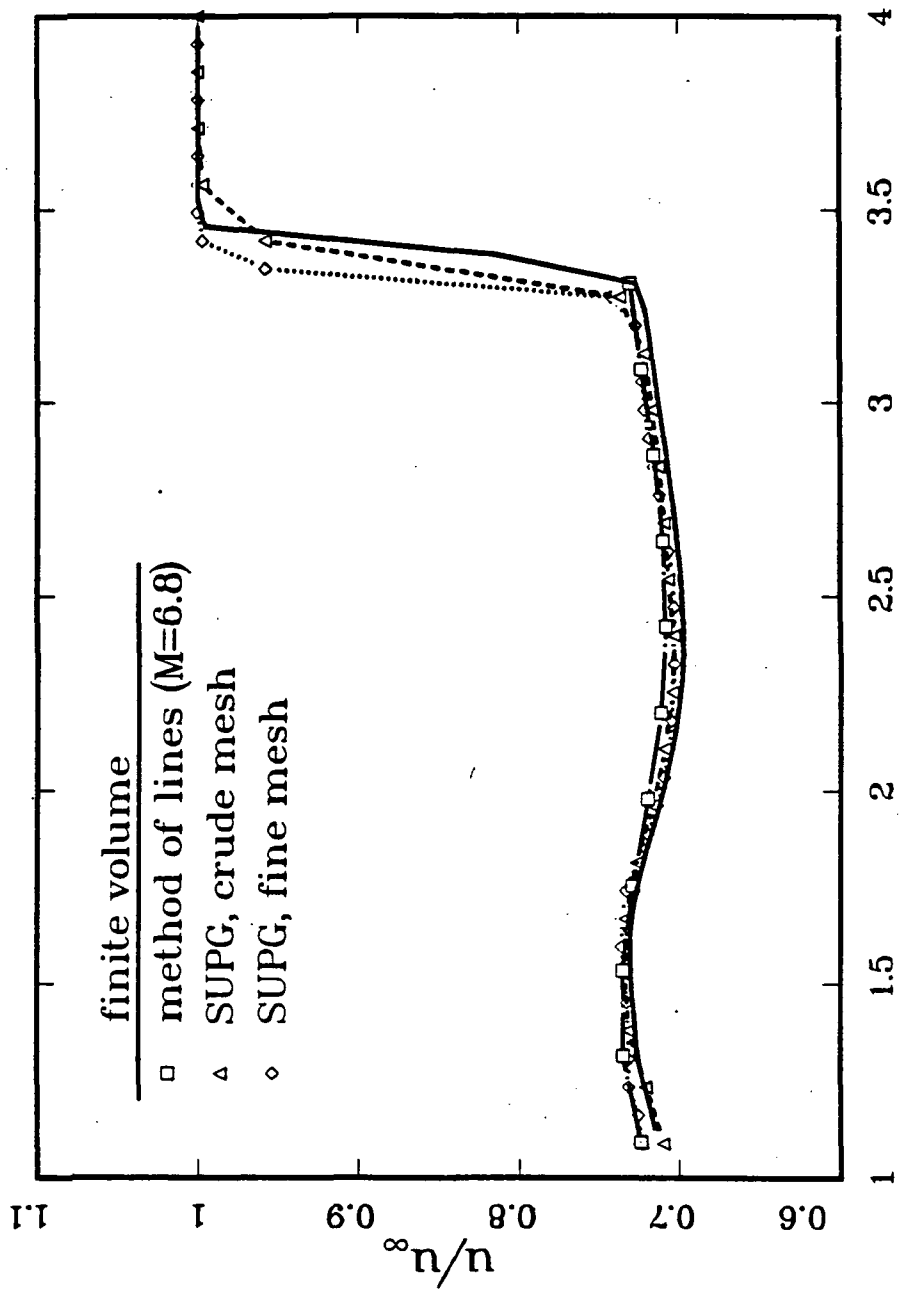


Fig. 4.23 Comparative velocity distributions at the outflow for the blunt leading edge using a modified Petrov-Galerkin formulation, Ref. [18].

finite volume method. The shock at the outflow is seen to be captured within 3 nodes without any post-shock oscillations.

The results of the modified Petrov-Galerkin method indicate the need to use this modification for compressible flow calculations, especially for flows where conservation of the advection flux is critical. The modifications to the Petrov-Galerkin algorithm do not seem to present any special hardships for vectorization. A preliminary investigation of the modified Petrov-Galerkin formulation indicates minor changes in the programming strategy and computational speed and a major improvement in solution quality and accuracy.

4.4 Evaluation of Programming Strategy

The merits of the vectorization procedures of Chapter 3 are highlighted by Table 4.1 which compares the computational speed for the scalar and vectorized version of the Taylor-Galerkin algorithm. Three flow problems, ranging from a problem with 100 elements to one with over 1000 elements, are used to quantify the effectiveness of the vectorization procedure. The scalar program was run on a CDC 855 and the vectorized version was run on the VPS-32. The figures in Table 4.1 indicate that the larger the number of elements in a flow analysis the bigger the computational benefits of using the vectorization procedure.

The performance evaluation of the finite element programs involves issues such as computational speed, storage requirements, and speed of convergence. Effective programming strategies on the VPS-32 include, in addition to vectorization schemes, factors such as efficient use of central memory, organization of data structures, and use of virtual memory.

C-2

Table 4.1

Comparison of Computational Rates for Scalar and Vectorized Versions of Taylor-Galerkin Algorithm

PROBLEM	CPU SECS		
N is the number of elements	SCALAR CODE CY 170-855	VECTOR CODE CYBER 205	SCALAR / VECTOR
Shock tube N = 100	234	3.5	68
Wedge N = 672	4047	14	280
Woodward Collela N = 1008	1962	6	330

The computational speed of a compressible flow program is usually rated as the processing time in CPU time per node per time-step. Typical computational speeds of the Taylor-Galerkin and Petrov-Galerkin formulations are:

Taylor-Galerkin - 3.3×10^{-5} CPUs/time-step/node

Petrov-Galerkin - 9.1×10^{-5} CPUs/time-step/node

The results show that the Taylor-Galerkin program is about three times faster than the Petrov-Galerkin program. The numerical integration procedure for the evaluation of the element matrices needed for the Petrov-Galerkin algorithm at each time-step is the reason for its higher computational times. Yet, the processing rate for the Petrov-Galerkin formulation with the four point Gauss integration is seen to be competitive with the Taylor-Galerkin procedure. If one point integration procedures could be developed for accurate integral evaluations, the computational speed of the two formulations would be comparable.

On the VPS-32 it is possible to use a timing package to obtain information regarding the CPU time spent in each subroutine or in specific operations within a subroutine. Information obtained from the timing package can be used to identify the most expensive operations in the formulation. Strategies can then be developed to improve the method of programming those expensive operations.

Figures 4.24 and 4.25 illustrate the flowchart for the Taylor-Galerkin and Petrov-Galerkin algorithms. The timing data for the main operations of both algorithms appear in Tables 4.2 and 4.3. For the

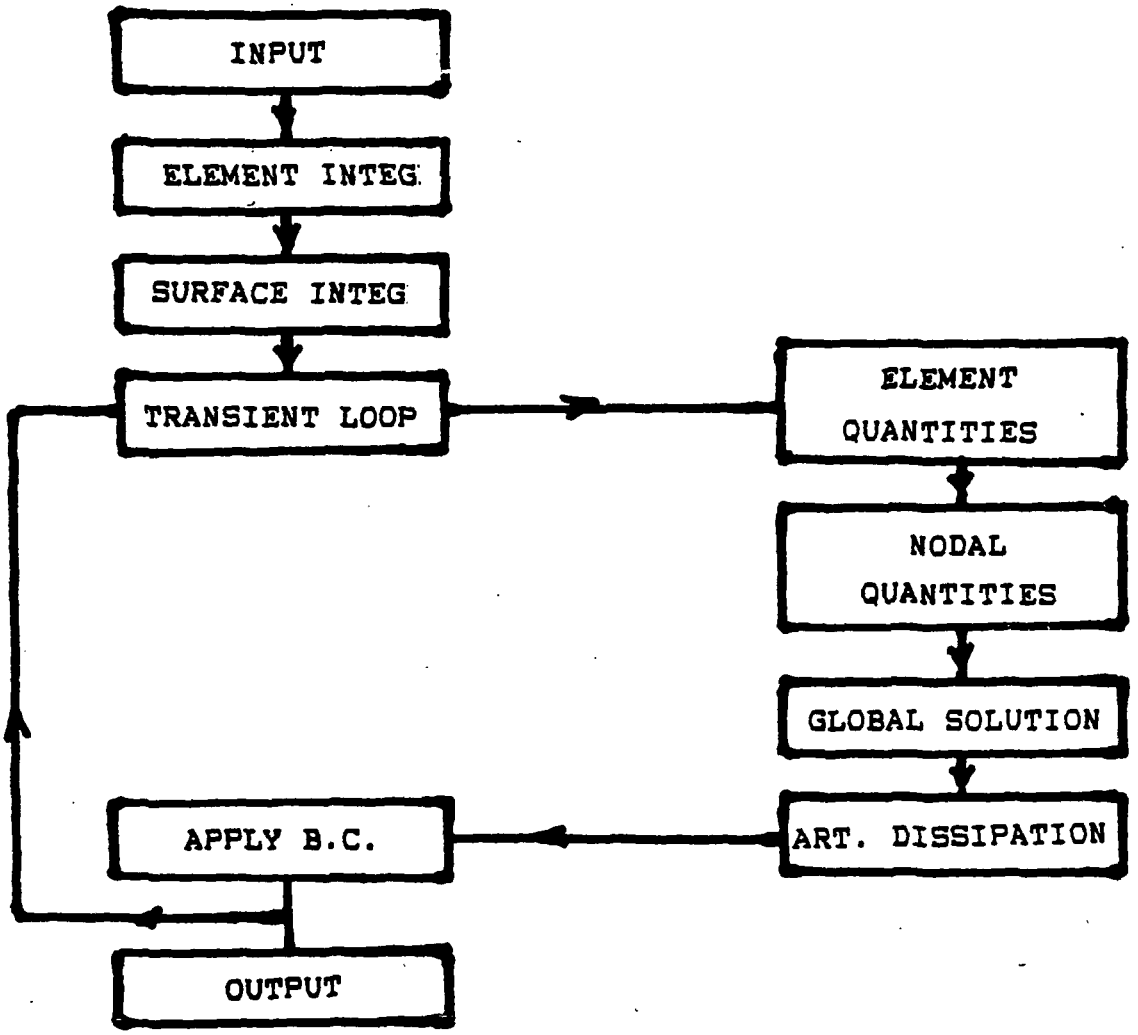


Fig. 4.24 Program flowchart for inviscid Taylor-Galerkin algorithm

Table 4.2

Timing Data for Principal Operations for Inviscid Taylor-Galerkin Algorithm

OPERATION	CPU TIME (%)
Input	1
Element integrals	0.5
Surface integrals	-
Half step calculations	13
Second step calculations	12
Artificial dissipation	56
Solution of global equations	6
Application of boundary conditions	1
Local time-step computations	.7

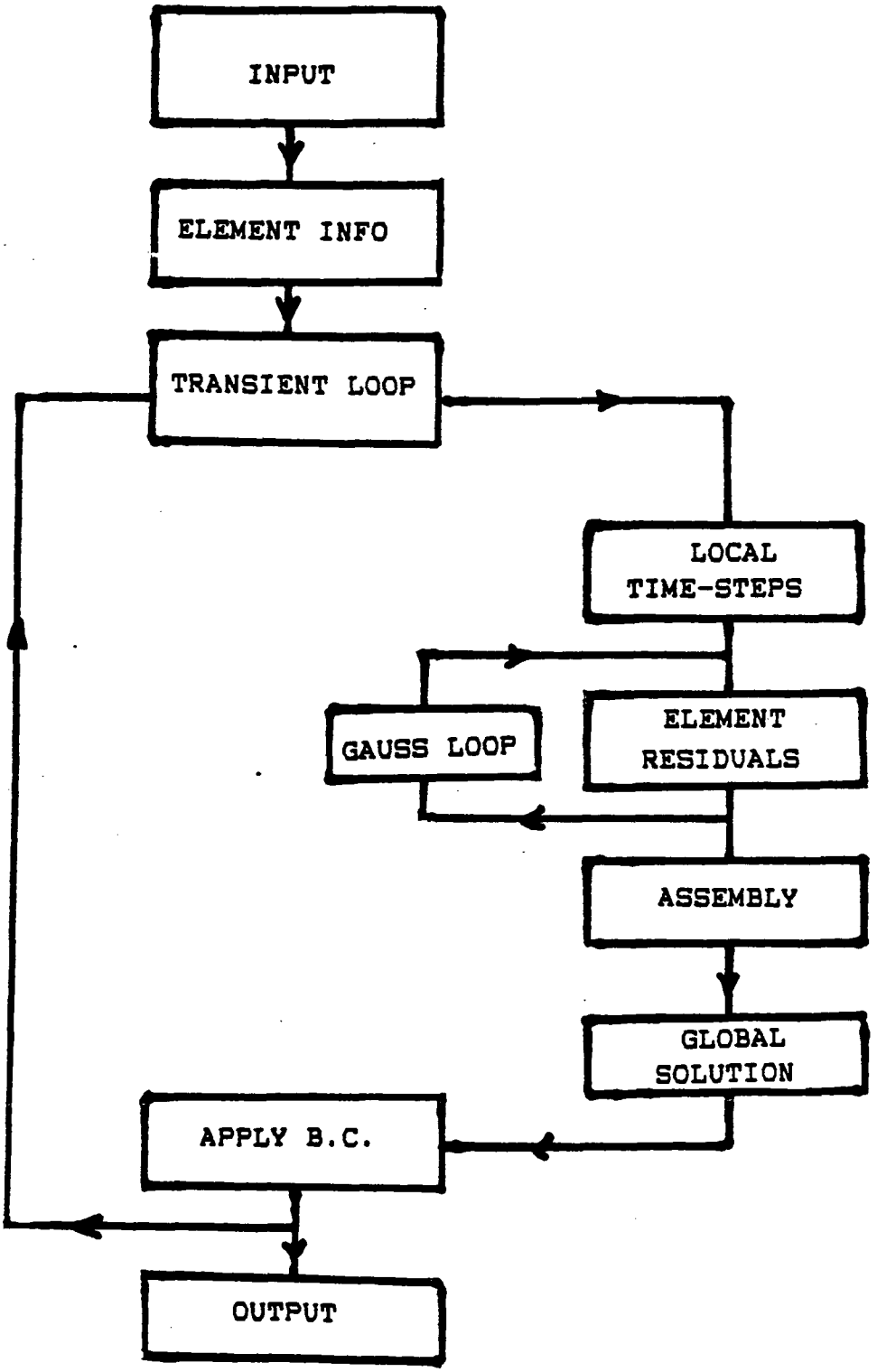


Fig. 4.25 Program flowchart for inviscid Petrov-Galerkin algorithm

Table 4.3

Timing Data for Principal Operations for Inviscid Petrov-Galerkin Algorithm

OPERATION	CPU TIME (%)
Input	1
Element information (element interpolation derivatives, etc.)	0.5
Computational of element residuals	86
Assembly of element residual matrices	1
Solution of global equations	4
Application of boundary conditions	1
Local time-step computations	6

Taylor-Galerkin formulation it is seen that the addition of artificial dissipation takes over 50% of the total CPU time. The element integrals that need to be evaluated for the artificial dissipation terms use four point Gauss integration which, even when fully vectorized, are seen to be computationally expensive. The timing information for the Petrov-Galerkin formulation indicates that major portions of the total CPU time is shared between the various terms needed in the evaluation of the element residual vectors.

On supercomputers, such as the VPS-32, programs are usually written to have all the working arrays stored in central memory. The capability of the two finite element programs to handle large problems can be gauged by comparing the central memory required to work a sample problem. For a problem containing 1800 elements and 1911 nodes the storage required for the two algorithms is :

	Words of memory	Large pages
Taylor-Galerkin	350,000	6
Petrov-Galerkin	325,000	5

These numbers show that the Petrov-Galerkin algorithm requires about 10% less storage than the Taylor-Galerkin algorithm. This difference can be attributed to the Taylor-Galerkin algorithm computing and storing all the element integrals outside of the time-step loop.

The finite element programs developed are pseudo-steady state codes wherein the steady state solutions are obtained by time marching. Convergence is assumed to occur when the L_2 norm of the

conservation variables drop over three orders of magnitude. For such time marching schemes fast convergence rates are essential. The faster a program converges the less the computational cost. The use of local time-stepping procedures to improve convergence was demonstrated earlier and both formulations were implemented with the time-stepping procedure of section 4.2. Experience gained working numerous inviscid problems indicate that the Taylor-Galerkin formulation can be run at higher values of the safety factor σ. Typically the Taylor-Galerkin formulations can run at values of σ over 0.6, while the Petrov-Galerkin formulations usually work at σ values of about 0.3. The trend is typical of upwind schemes which need lower safety factors than the explicit artificial dissipation schemes.

The rate of convergence for both formulations can be compared by plotting the L_2 norm of the changes in the conservation variables. For Mach 3 flow over the compression corner, the L_2 norm of the changes in the regular conservation variables for the Taylor-Galerkin algorithm and in the entropy variables for the Petrov-Galerkin algorithm appear in Figs. 4.26 and 4.27. The figures indicate that all the conservation variables follow similar trends regarding the rates of convergence, and the Petrov-Galerkin algorithm is seen to converge an order of magnitude more than the Taylor-Galerkin algorithm.

A realistic measure of the convergence rates can be obtained by comparing the rate of convergence of the total pressure force on a body surface. For the compression corner the pressure force on the inclined wall can be computed exactly since the flow parameters after the shock can be obtained from the oblique shock relations. Figure 4.28 plots the time history for the total pressure force on the inclined wall for

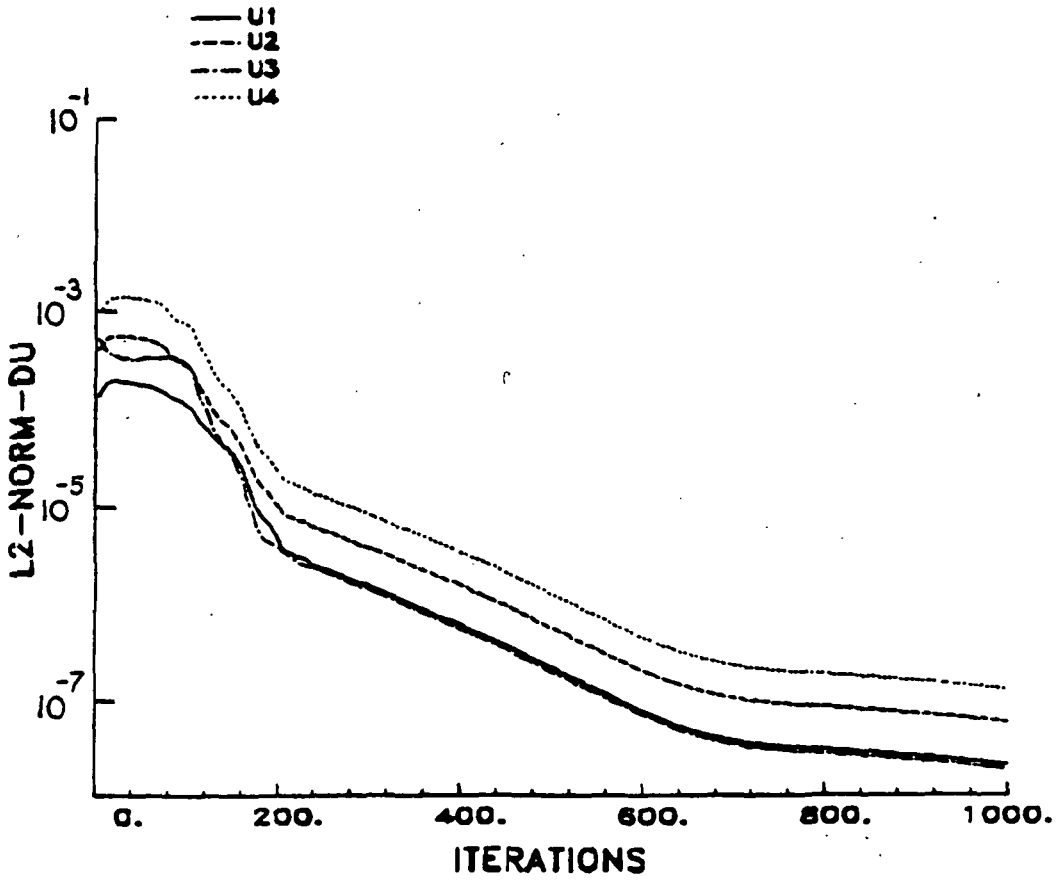


Fig. 4.26 Convergence rates for conservation variables for compression corner using Taylor-Galerkin algorithm.

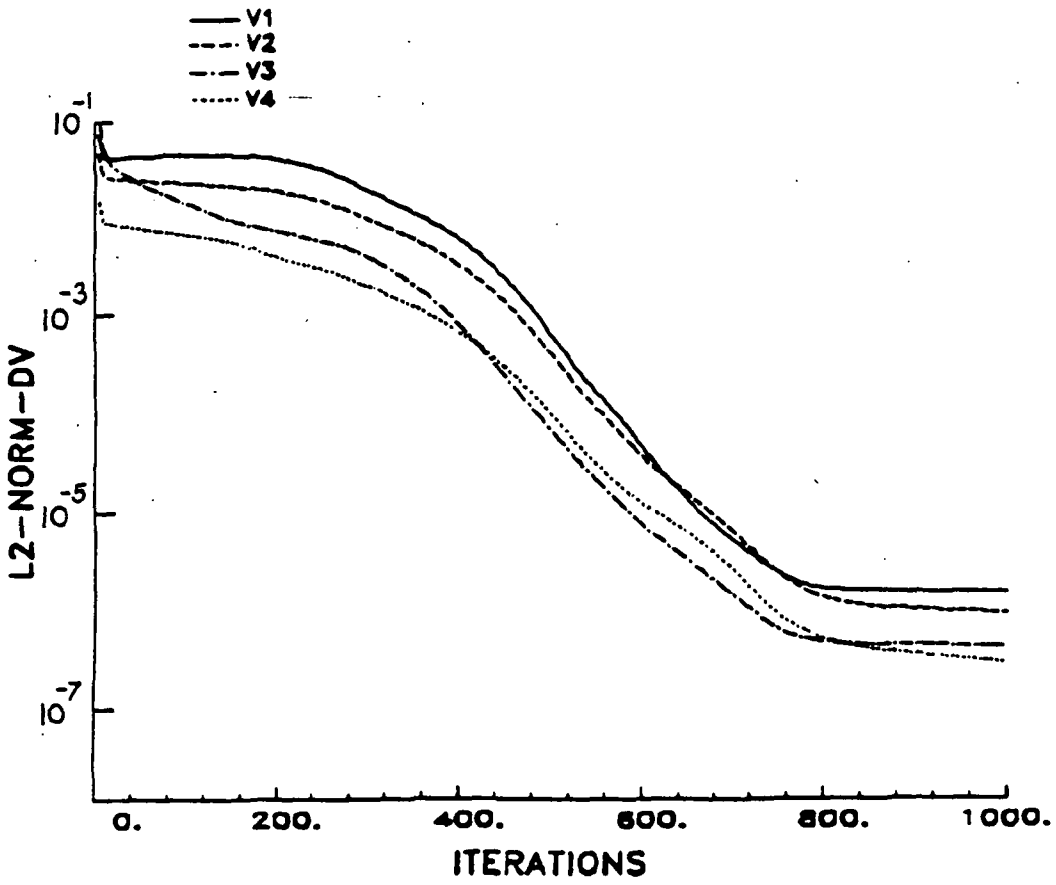


Fig. 4.27 Convergence rates for entropy variables for compression corner using Petrov-Galerkin algorithm.

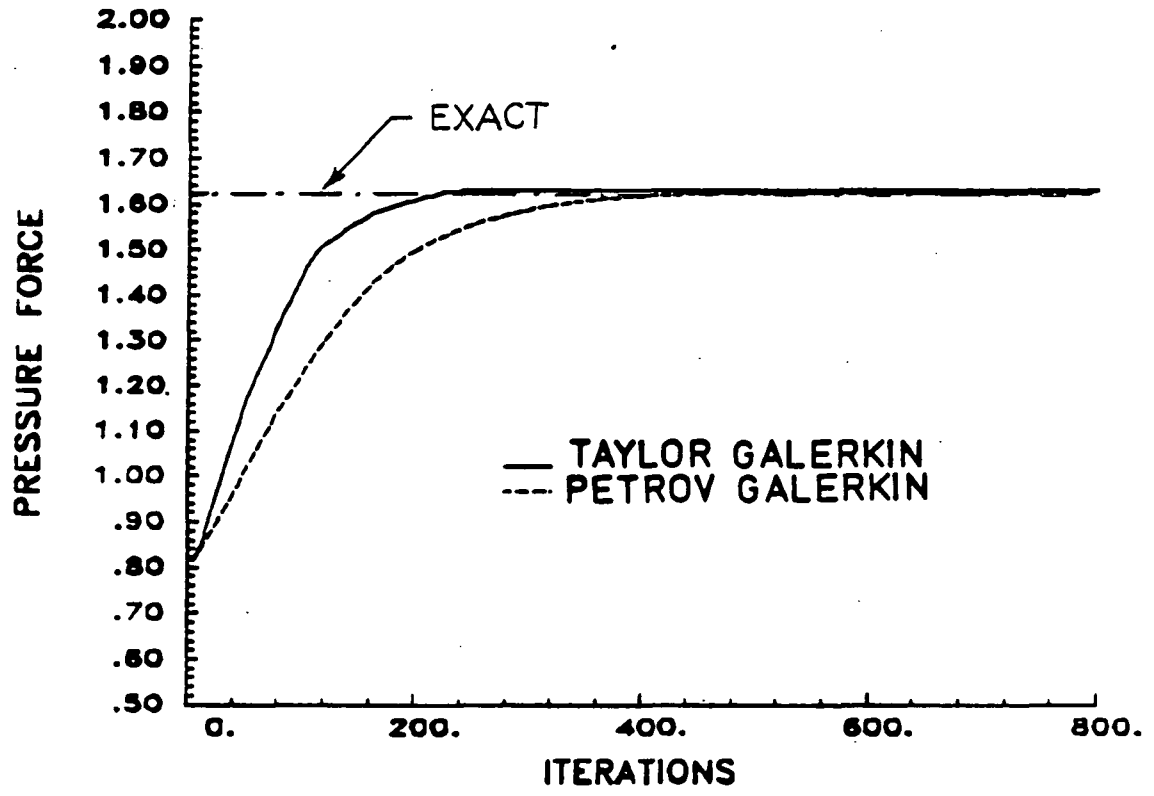


Fig. 4.28 Comparative convergence of finite element algorithms based on pressure force on compression corner wall.

the Taylor-Galerkin and the Petrov-Galerkin algorithms. The pressure force computed using the exact shock relations is also indicated in Fig. 4.28. The Taylor-Galerkin algorithm reaches the value of the pressure force given by the exact solution in about 250 iterations while the Petrov-Galerkin needs about 500 iterations to reach the same pressure force value. The fast convergence for the Taylor-Galerkin algorithm can be attributed to the higher safety factor σ used for computations.

The convergence trends implied by Figs. 4.26 and 4.27 are at odds with those indicated by Fig. 4.28. The use of the pressure force is seen to be a better indicator for convergence rates than the L_2 norm of the change in conservation and entropy variables. The use of total pressure force for inviscid flows and shear or heat fluxes for viscous flows for evaluating convergence rates merits further investigation.

4.5 Closing Comments

The vectorization strategies of Chap. 3 have been applied to the Taylor-Galerkin and Petrov-Galerkin algorithms for 2D inviscid flows. This chapter has presented comparative results for the two algorithms. The algorithms have basic differences in philosophy which include the mode of artificial dissipation and the method of integral evaluations. The vectorization of the algorithms resulted in programs of comparable computational speeds and storage. The vectorization strategies used for the 2D inviscid flows are extended to 2D viscous flows in the next chapter.

Chapter 5

VISCOUS 2D COMPUTATIONS

In the realistic problems encountered in high speed compressible flows, the flow behavior is influenced a great deal by viscous effects. For supersonic and hypersonic flight vehicles the effects of viscous heat dissipation, and shock-viscous interactions may play a crucial role in the performance of the vehicle.

The flow features for high Reynolds number compressible flows can be predicted by two approaches. The first approach is to divide the flowfield into an inviscid region and a viscous region. A matching of the viscous boundary conditions enables the coupling of the viscid and inviscid regions. This approach reduces computational costs but is limited in applications to problems not involving effects such as flow separation or shock-boundary layer interactions [48]. The second approach is a global approach, wherein the Navier-Stokes equations, which are valid throughout the entire domain, are used to predict flow details everywhere. This approach is relatively straightforward but computationally difficult; however, the approach is necessary for complicated viscous problems such as shock-boundary layer interactions. In this dissertation a finite element Petrov-Galerkin formulation for the compressible Navier-Stokes equations is used to predict 2D viscous flow characteristics.

5.1 Navier-Stokes equations

The compressible Navier-Stokes equations describe the characteristics of a laminar, compressible, viscous, heat conducting fluid and can be written as,

$$U_{,t} + F_{i,i} = F^v_{i,i} + F^h_{i,i} \tag{5.1}$$

where U is the vector of conservation variables and F^v_i, F^v_i, and F^h_i are the fluxes that correspond to the advection, viscous dissipation, and heat diffusion respectively. The vector U and the flux vectors in two dimensions are given by,

$$U = \rho \begin{Bmatrix} 1 \\ u_1 \\ u_2 \\ E_t \end{Bmatrix} \qquad F_i = u_i U + p \begin{Bmatrix} 0 \\ \delta_{1i} \\ \delta_{2i} \\ u_i \end{Bmatrix}$$

$$F^v_i = \begin{Bmatrix} 0 \\ \tau_{i1} \\ \tau_{i2} \\ \tau_{ij} u_j \end{Bmatrix} \qquad F^h_i = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ q_i \end{Bmatrix} \tag{5.2}$$

Here ρ is the density, p the pressure, u_i the velocities in the coordinate directions, q_i the heat fluxes, τ_{ij} the viscous stress components and E_t the total energy. The Kronecker delta δ_{ij} is as defined in Eq. (2.14). The shear stresses and the heat fluxes are given by,

$$\tau_{ij} = \lambda\, u_{k,k}\, \delta_{ij} + \mu\, (u_{i,j} + u_{j,i}) \tag{5.3}$$

$$q_i = -k\, T_{,i} \tag{5.4}$$

where μ and λ are the coefficients of viscosity, k the thermal conductivity, and T is the temperature.

The Petrov-Galerkin formulation uses entropy functions to symmetrize the Navier-Stokes equations. The advantage of using a symmetric system of equations is that a weighted residual formulation based on these equations automatically inherits the stability possessed by the exact solution of these equations [14].

The spatial derivatives of the inviscid, viscous, and heat fluxes can be written as,

$$F_{i,i} = F_{i,U}\, U_{,i} = A_i U_{,i} \tag{5.5}$$

$$F_i^v = K_{ij}^v\, U_{,j} \tag{5.6}$$

$$F_i^h = K_{ij}^h\, U_{,j} \tag{5.7}$$

Using the above relations in Eq. (5.1) the Navier-Stokes equations become,

$$U_{,t} + A_i U_{,i} = (K_{ij}^v + K_{ij}^h)_{,i} \tag{5.8}$$

A change of variables is introduced by defining new independent variables V to replace the conservation variables U. A one-to-one mapping is assumed between U and V and Eq. (5.8) transforms to,

$$A_o\, V_{,t} + \tilde{A}_i\, V_{,i} = (\tilde{K}_{ij}\, V_{,j})_{,i} \tag{5.9}$$

where

$$A_o = U_{,v} \tag{5.10}$$

$$\tilde{A}_i = A_i A_o \tag{5.11}$$

$$\tilde{K}_{ij} = K_{ij} A_o \tag{5.12}$$

The matrices A_o, $\tilde{A}_i$ and $\tilde{K}_{ij}$ are symmetric, and A_o and $\tilde{K}_{ij}$ are positive definite. The terms of these matrices are given in [39]. The definition of V is based on the entropy functions and is given by Eqs. (2.47)-(2.51).

5.2 Finite element formulation

The weighted residual formulation for (5.9) is given by,

$$\int_A W^T \left(A_0 V_{,t} + \tilde{A}_i V_{,i} - (\tilde{K}_{ij} V_{,j})_{,i}\right) dA = 0 \qquad (5.13)$$

where W is a matrix of weighting functions different from the shape functions [N] that appear in Eq. (2.12). The weighting function for the viscous formulation is based on the weighting function used for the inviscid formulation Eq. (2.34). The modification in W in Eq. (5.13) is the need to gauge the relative importance of the artificial and viscous diffusion in specific regions of the flow. The weighting functions W for the streamline upwind part of the Petrov-Galerkin formulation is given by Eq. (2.54) as,

$$W = N + (\tilde{A}_i \; T|\Lambda|^{-1}T^T)^T N_{,i} \quad . \qquad (5.14)$$

where the matrix T depends on the eigenvalues of the Jacobian matrices $\tilde{A}_i$ as well as on the local Peclet number. The Peclet number is a measure of the relative importance of the convective and diffusive effects and is defined as,

$$Pe = \frac{|\lambda_i| h}{2k} \qquad (5.15)$$

where λ_i is defined by the eigenvalue problem for T_i, and k is the thermal diffusivity. In the boundary layer, the local Peclet number is small, and viscous diffusion terms predominate. In the inviscid dominated regions, the local Peclet number is very large (advection dominates), and the numerical diffusion terms come into play. The use of a doubly asymptotic function of the local Peclet number to limit the

effects of numerical dissipation in the boundary layer is discussed in [39] and is used here for the viscous calculations that follow.

5.3 Sample Problem

The performance of the vectorized Petrov-Galerkin formulation is illustrated using the steady viscous supersonic flow past an isothermal flat plate as a test problem, Carter [49]. The solution domain and the discretization for this problem appear in Fig. 5.1. The finite element mesh for the problem contains 3111 nodes and 3000 elements. The mesh is graded near the leading edge to resolve the leading edge effects. The inflow is supersonic at Mach 3, and a Reynolds number of 1000 based on the length of the flat plate is assumed at inflow. The viscosity of the fluid was assumed to vary according to the Sutherland law, and the thermal conductivity was obtained from a constant Prandtl number taken to be 0.72. Figure 5.2 shows the density contours for the flowfield at convergence using the Petrov-Galerkin algorithm. The presence of the leading edge shock and the boundary layer is clearly visible from the contours. The results obtained from the Petrov-Galerkin formulation are compared to those of Carter [49] and Taylor-Galerkin results, [50]. Carter's mesh was 45x50 and the mesh used by the Taylor-Galerkin algorithm was 66x51. The distribution of density and velocity at the outflow for the three methods appears in Figs. 5.3a - 5.3c. The density and u-velocity distribution for the three methods compare well. The Petrov-Galerkin distributions for the v-velocity show a kink at the wall but away from the wall the distributions agree well with those of Carter and the Taylor-Galerkin algorithm. The discrepancy at

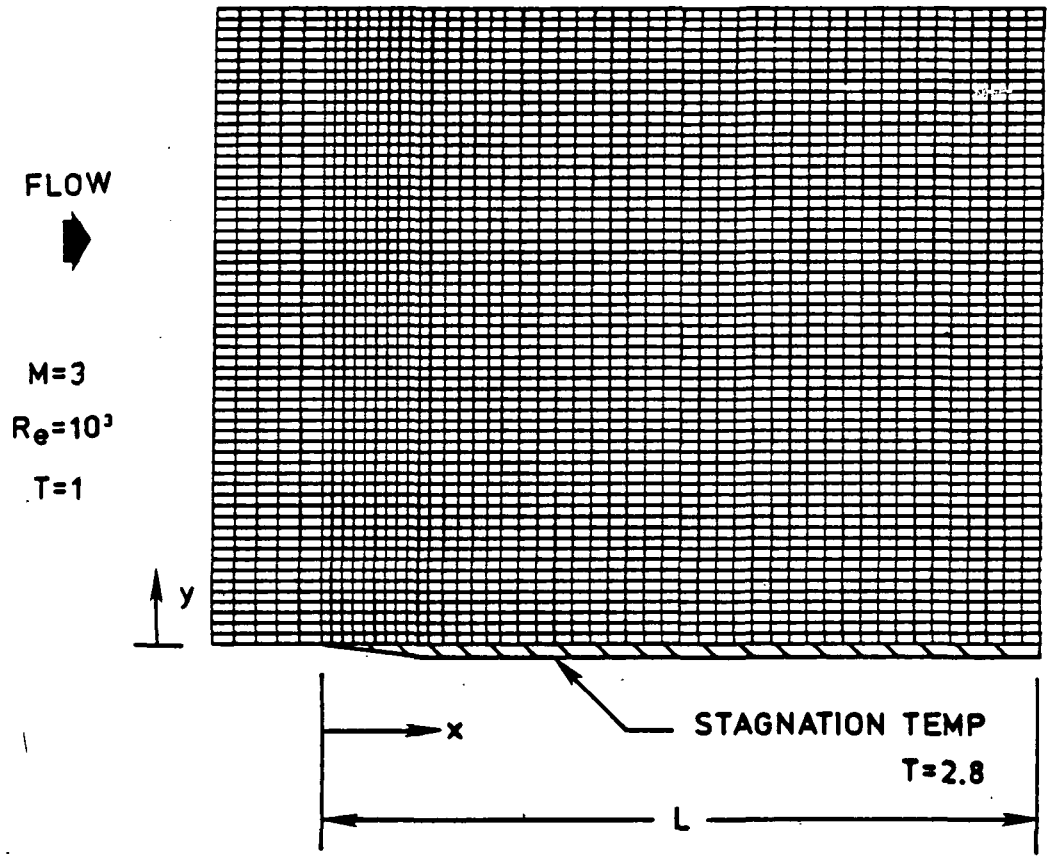


Fig. 5.1 Flow configuration and finite element mesh for Mach 3 flow over flat plate.

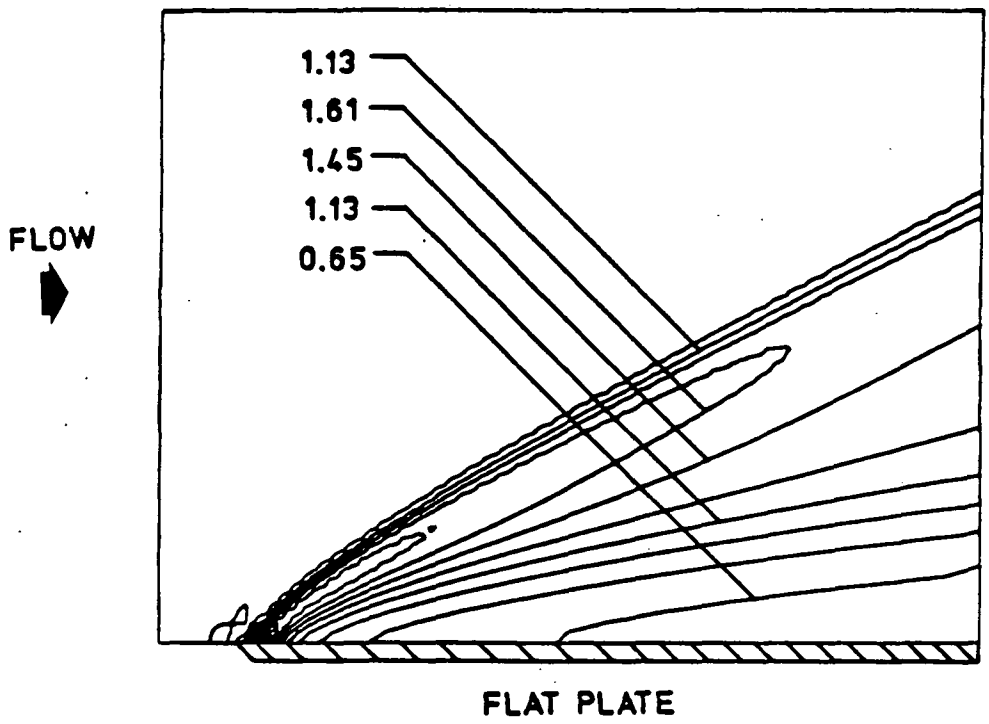
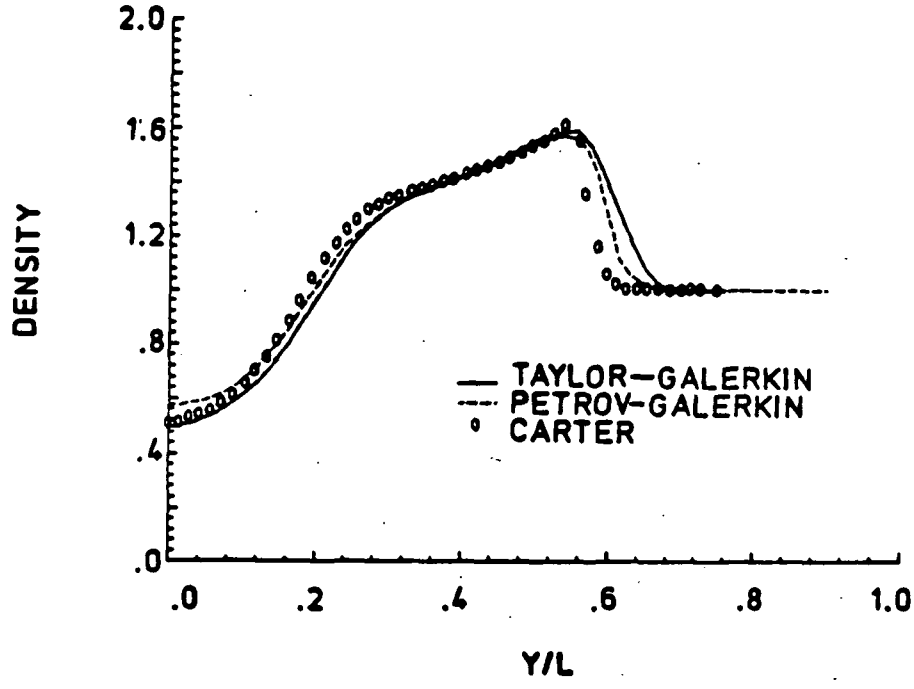
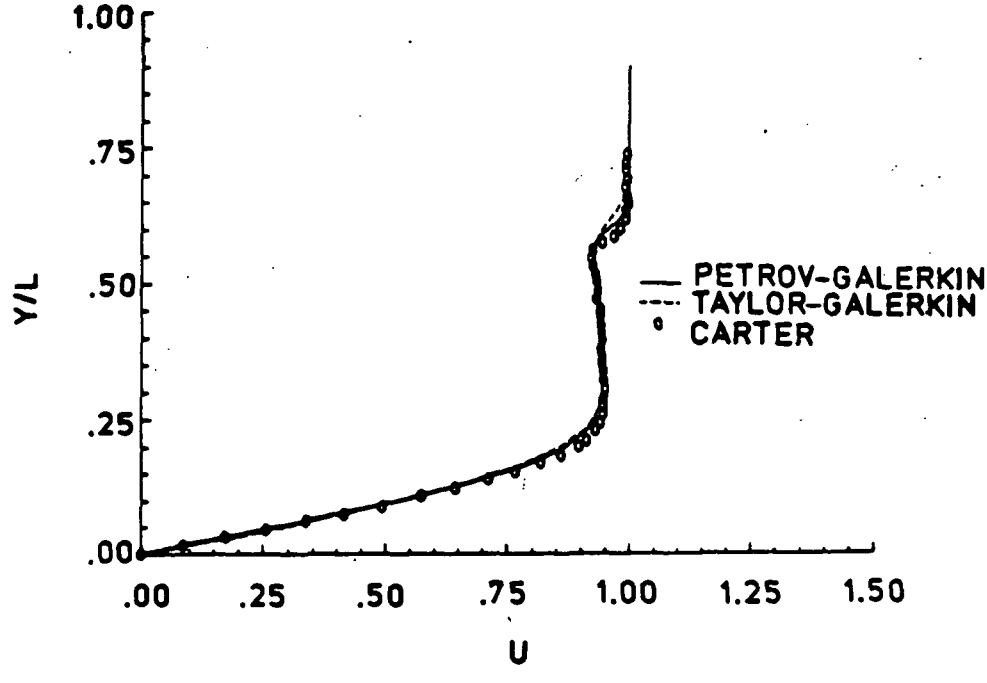


Fig. 5.2 Density contours for flow over a flat plate using the Petrov-Galerkin algorithm.

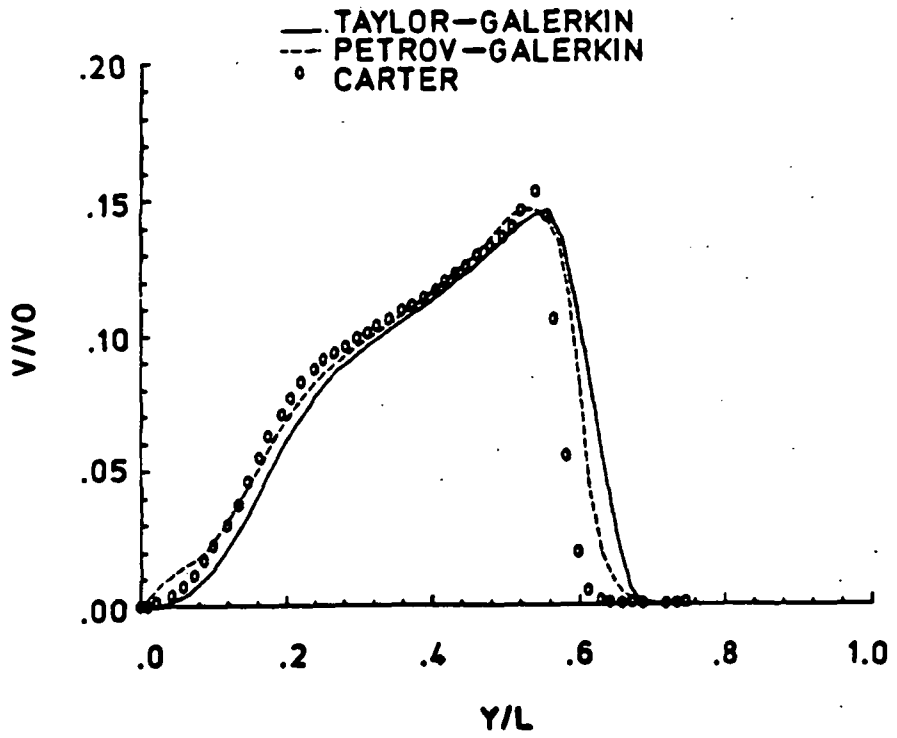


(a) Comparative density distributions at exit

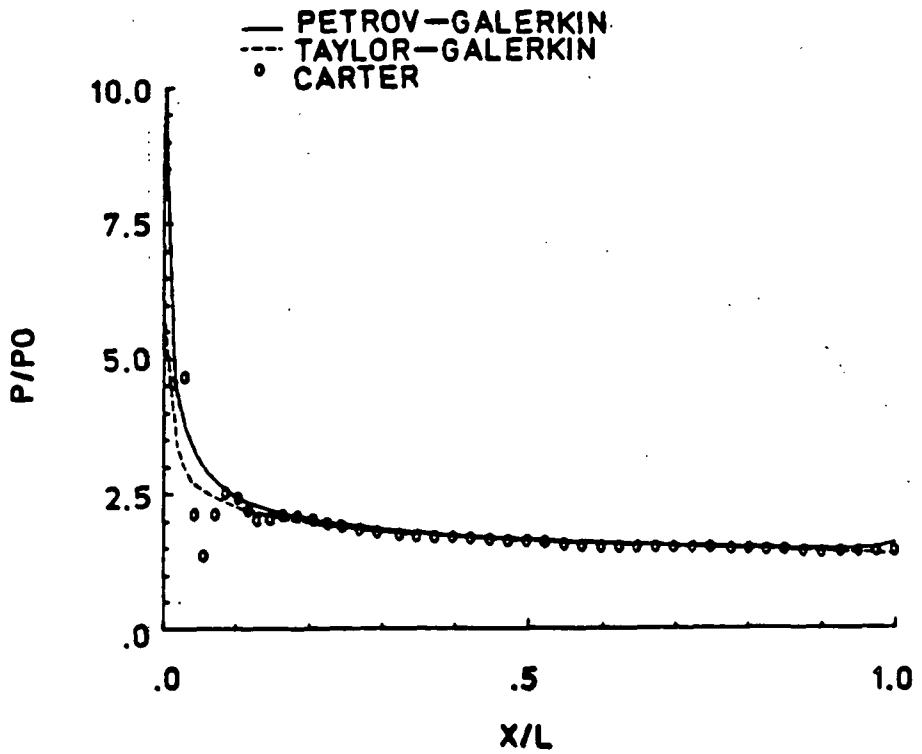


(b) Comparative u-velocity distributions at exit

Fig. 5.3 Comparative distributions at the wall and at the outflow for flow over flat plate.



(c) Comparative v-velocity distributions at exit



(d) Comparative pressures distributions along wall

Fig. 5.3 Comparative distributions at the wall and at the outflow for flow over flat plate.

the wall is due to the Petrov-Galerkin formulation being sensitive to the outflow conditions close to the wall where the flow is subsonic. Fig. 5.3d shows the distribution of pressure along the plate for the three methods. The Petrov-Galerkin and Taylor-Galerkin algorithms show a smooth variation along the flat plate while the pressures predicted by Carter show a few oscillations at the leading edge. The plots indicate the validity of the computing procedure for 2D viscous flows.

5.4 Comments on 2D Viscous Program

As with the inviscid terms, the viscous terms in the Petrov-Galerkin formulation are nonlinear and need to be evaluated using Gauss quadrature. The vectorization strategies used to vectorize the inviscid terms can be extended to the integrals that arise from the addition of the viscous terms. The assemblage of the residual vectors due to the viscous terms and the solution of the system equations is similar to the procedures outlined in Chap. 3.

Figure 5.4 details the program flowchart for the 2D Petrov-Galerkin viscous formulation. The CPU seconds required for the main operations in the formulation appear in Table 5.1. Computation of the viscous terms takes a relatively small fraction of the total time (15 %), with an increase in storage of less than 5%. Typical computational speeds of the 2D viscous Taylor-Galerkin and Petrov-Galerkin algorithms are:

Taylor-Galerkin - 4.5×10^{-5} CPUs/time-step/node

Petrov-Galerkin - 10.5×10^{-5} CPUs/time-step/node

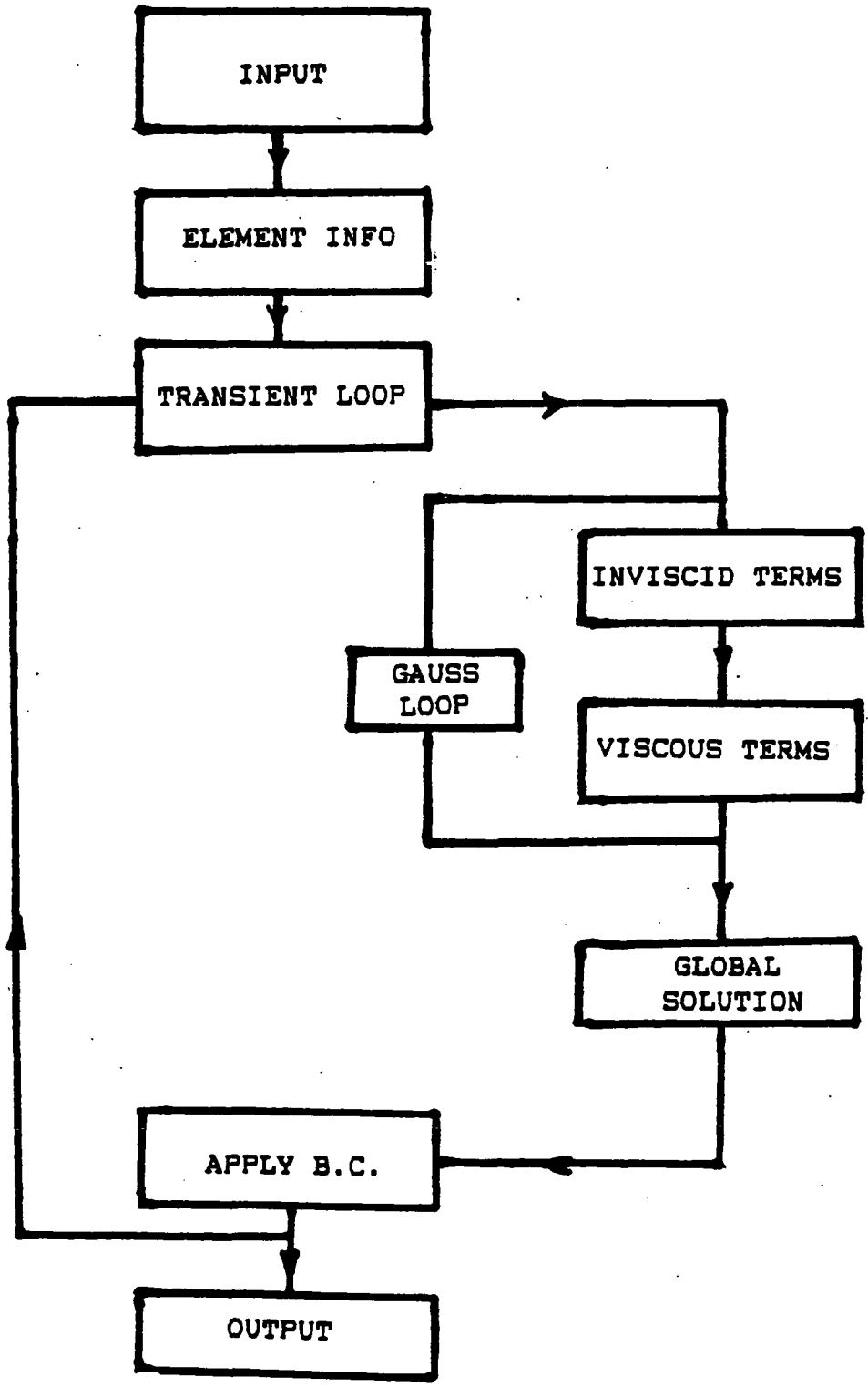


Fig. 5.4 Program flowchart for viscous Petrov-Galerkin algorithm.

Table 5.1

Timing Data for Principal Operations for Viscous Petrov-Galerkin Algorithm

OPERATION	CPU TIME (%)
Input	1
Element information (element interpolation derivatives, etc.)	0.5
Inviscid terms	80
Viscous terms	13
Solution of global equations	4
Application of boundary conditions	1

The storage needed for the two algorithms can be compared by comparing the total arrays, both real and integer, needed to work typical problems. Storage requirements for the two formulations are given by,

Taylor-Galerkin - 60 x NPOIN + 336 x NELEM

Petrov-Galerkin - 13 x NPOIN + 170 x NELEM

where NPOIN and NELEM are the number of nodes and elements in the domain. The figures indicate that the Petrov-Galerkin formulation uses less than 50% of the storage needed for the Taylor-Galerkin formulation. The element integrals for the inviscid and viscous terms are calculated and stored before the start of the transient loop for the Taylor-Galerkin, and thus the need for a substantially larger storage.

A drawback of the Petrov-Galerkin viscous formulation, as mentioned earlier, is the need for numerical integration. For 2D problems the use of 2x2 Gauss quadrature is seen to be adequate. This implies the need for 2x2x2 integration procedures for 3D analysis. In the vectorized program, the Gauss integration procedure needs to store the derivatives of the shape functions at the nodes for all elements at each Gauss point. For 3D problems this results in prohibitive storage requirements. Thus the development of accurate one point reduced Gauss integration will be necessary before the Petrov-Galerkin algorithm can be extended for 3D viscous flows.

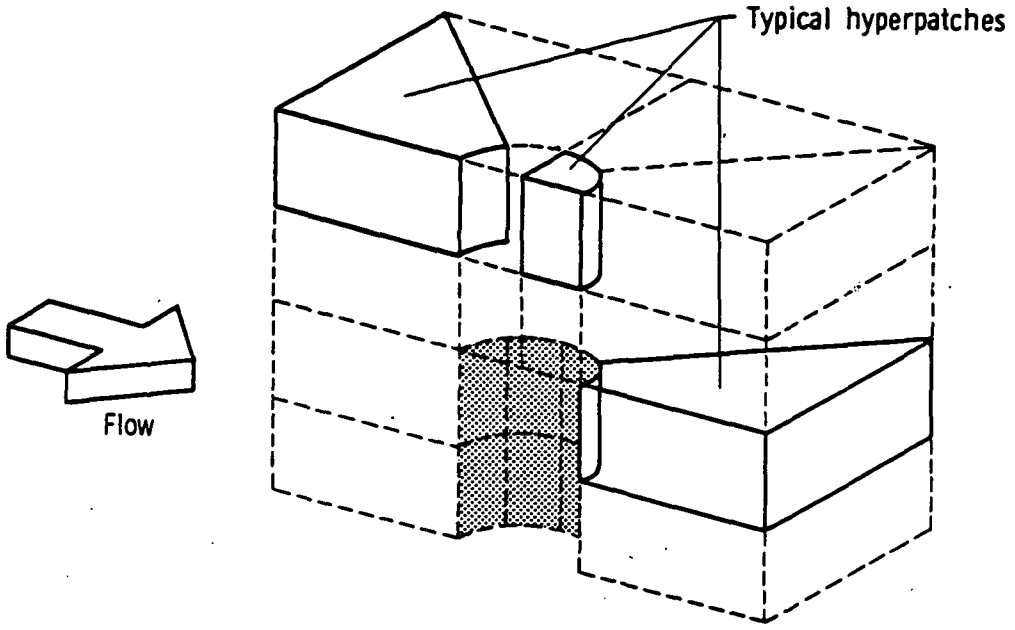
Chapter 6

COMPUTATIONS FOR 3D INVISCID FLOWS

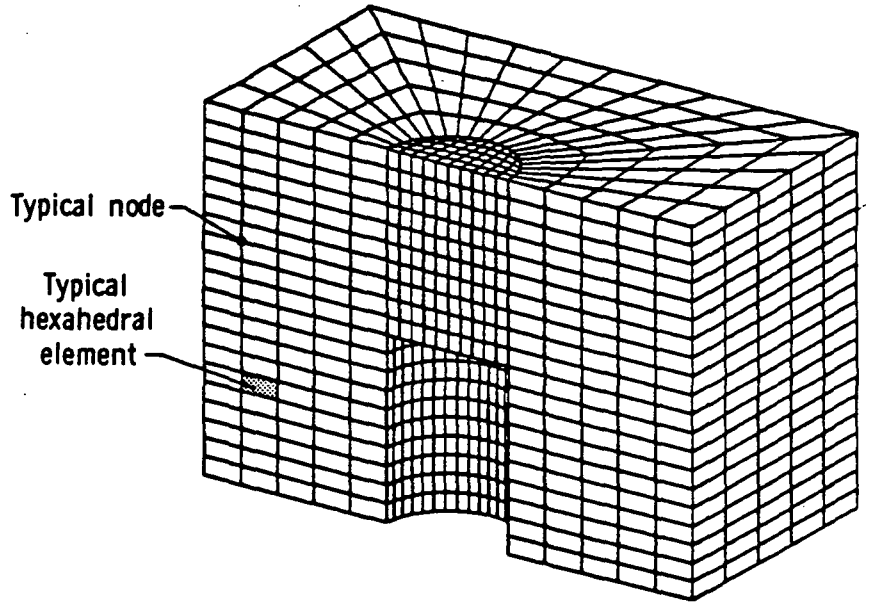
The use of vectorization strategies in simulating 2D compressible flow situations showed significant computational benefits. Vectorization is highly desirable for 2D flows but is essential for detailing 3D inviscid and viscous flows. To gain a further understanding of the role of vectorization in 3D, the strategies developed in Chap. 3 were used to implement the Taylor-Galerkin algorithm for three dimensional inviscid flow. Model generation procedures, display of results, details of the finite element formulation, and quality of solutions obtained are discussed in this chapter.

6.1 Model Generation and Results Display

The commercially available PATRAN program [19] is used extensively for finite element structural analysis. Modelling features of PATRAN can be exploited for fluid flow analysis. Modelling compressible fluid flow with PATRAN begins with the creation of a geometric representation of the computational domain. A 3D computational domain is shown in Fig. 6.1 and is represented by solid regions called hyperpatches. Finite element meshes are created by subdividing the hyperpatches into hexahedron or tetrahedron elements. The use of these elements depends on the capability of the analysis program. In the problems that follow hexahedron elements are used exclusively.



(a) Hyperpatch representation



(b) Finite element mesh

Fig. 6.1 Computational domain for typical flow problem.

Translator and inverse translator programs [19] have been developed at NASA Langley to interface PATRAN with flow analysis programs. The translator program uses output files generated by PATRAN to produce input data for a finite element analysis program. The results obtained from the analysis program are converted into the format required by PATRAN for results display by the inverse translator program.

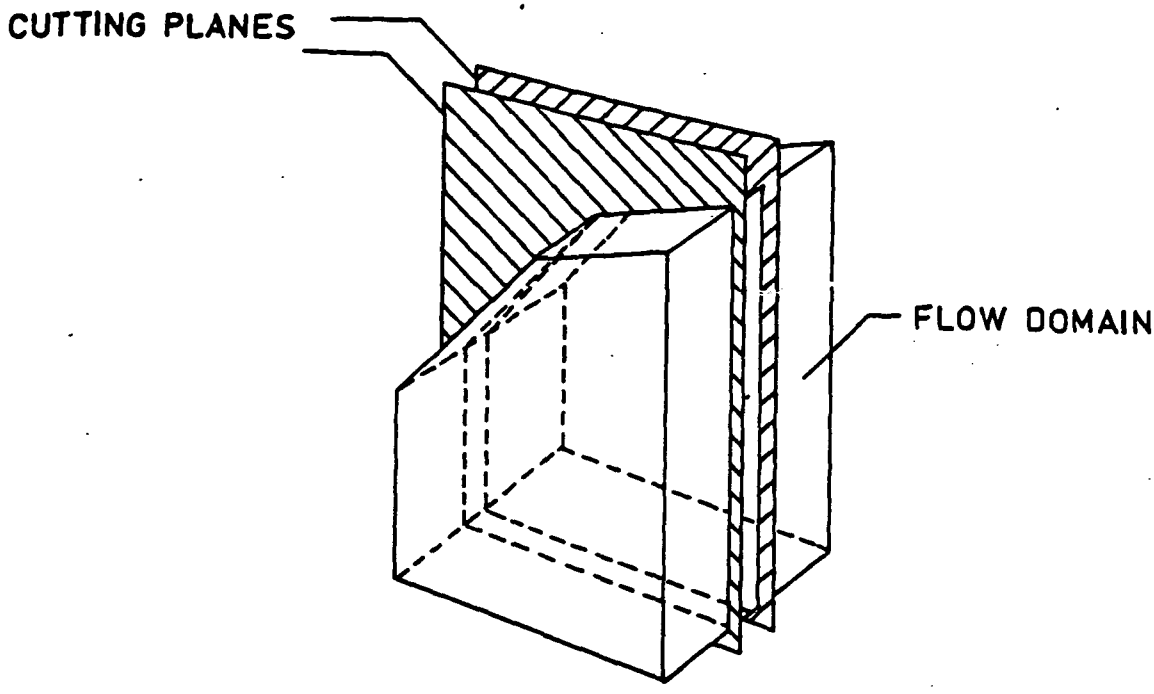
PATRAN has extensive capability to display scalar solution results with contour plots. Contours may be curved lines which connect points of constant value or solid, color-filled bands which define the limits of certain ranges of the quantity being displayed. Regions of interest in the model (e.g. surfaces, symmetry planes, outflow planes, etc.) are identified and elements in these regions are grouped together in "active sets." Active sets are smaller than full models, containing a few hundred elements and can be used effectively to display the salient characteristics of the entire flow. Figure 6.2 illustrates the concept of an active set and results display for a typical active set.

6.2 Taylor-Galerkin Algorithm

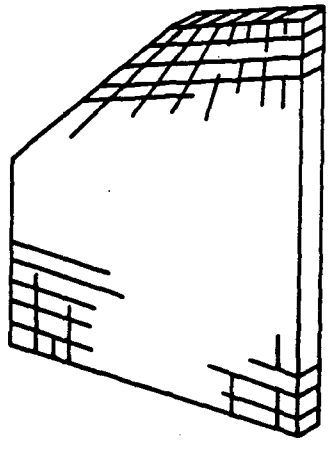
The Taylor-Galerkin algorithm was introduced in Chap. 2 and the extension of this formulation to three dimensions is straightforward, but key equations are repeated here for reference.

The 3D Euler equations in conservation form are given by,

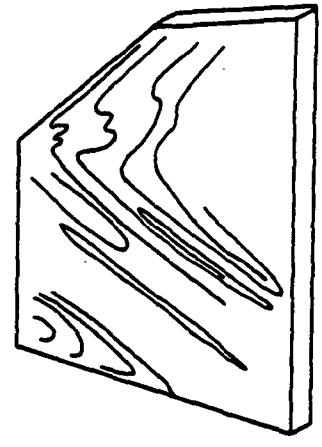
$$U_{,t} + F_{i,i} = 0 \tag{6.1}$$



(a) Cutting planes through flow domain



(b) Active set elements



(c) Contours displayed on active set elements

Fig. 6.2 Active set creation and display.

where U is the vector of conservation variables, and F_i are the flux vectors of mass, momentum and energy in the coordinate directions. The vector of conservation variables, U, and the flux vectors F_i are given by,

$$U = \rho \begin{Bmatrix} 1 \\ u_1 \\ u_2 \\ u_3 \\ E_t \end{Bmatrix} \qquad F_i = u_i U + p \begin{Bmatrix} 0 \\ \delta_{1i} \\ \delta_{2i} \\ \delta_{3i} \\ u_i \end{Bmatrix} \tag{6.2}$$

where ρ is the density, u_i the velocity components in the x, y, and z directions, and E_t is the total energy. The pressure p is defined as,

$$p = (\gamma - 1)\, \rho\, [E_t - 0.5(u_i u_i)] \tag{6.3}$$

Equation (6.1) is solved subject to proper initial and boundary conditions. The Taylor-Galerkin formulation is easier to derive by considering just one variable. For a typical variable u Eq. (6.1) can be written as,

$$u_{,t} + F_{i,i} = 0 \tag{6.4}$$

The computation proceeds through two time levels $t_{n+1/2}$ and t_{n+1}. At time level $t_{n+1/2}$, values for u are constant within an element while at time t_{n+1}, these constant element values are used to compute nodal values of u.

Time level $t_{n+1/2}$:

The constant element value $u_d^{n+1/2}$ is computed from,

$$V\, u_d^{n+1/2} = \int_V [N]\, dV\, \{u\}^n - \frac{\Delta t}{2} \int_V [N_{,i}]\, dV\, \{F_i\}^n \tag{6.5}$$

where V denotes an element volume, Δt is the time-step, and [N] is the matrix of element interpolation functions. On the outflow surfaces the weighted residual equation is written as,

$$A\, u_s^{n+1/2} = \int_A [N_s]\, dA\, \{u\}^n - \frac{\Delta t}{2} \int_A [N_s]\, dA\, \{F_{i,i}\}^n \tag{6.6}$$

In the above, $[N_s]$ denotes the interpolation functions of the gradients of the flux components on the outflow surfaces, and A is the outflow surface area. The element and surface quantities at $t_{n+1/2}$ are used to obtain the nodal values at time t_{n+1}.

Time level t_{n+1}:

An approximation to the Taylor series expansion of u at t_{n+1} and the application of the weighted residual statement on the resulting equation yields,

$$[M]\{u\}^{n+1} = [M]\{u\}^n + \Delta t \int_V [N_{,i}]\, dV\, E_i^{n+1/2} + \{R\}^{n+1/2} \tag{6.7}$$

where [M] is the element consistent mass matrix given by,

$$[M] = \int_V [N]^T [N]\, dV \tag{6.8}$$

and the load vector {R} is given by,

$$\{R\}^{n+1/2} = -\Delta t \int_A \ell_i\, E_{si}^{n+1/2}\, [N]\, dA \tag{6.9}$$

where ℓ_i are the components of the unit normal surface vector $\hat{n}$. The flux components on the surface, $E_{si}^{n+1/2}$ are obtained using the surface quantities, $u_s^{n+1/2}$, computed at the half step.

To handle flows with sharp gradients such as shocks, artificial dissipation is added at the end of each timestep. The 3D formulation uses Lapidus dissipation Eq. (2.30), and the addition of dissipation is of the form,

$$u_s^{n+1} = u^{n+1} + \nu \, \Delta t \, E_{i,i} \tag{6.10}$$

where,

$$E_i = h^2 \left| u_{i,i} \right| u_{,i} \qquad (i \text{ not summed}) \tag{6.11}$$

where ν is the Lapidus coefficient, Δt the timestep, and h a characteristic element length.

The element integrals that appear in Eqs. (6.5)-(6.9) were evaluated using closed form integration. The use of numerical integration in three dimensions would result in inflated storage requirements and increased computational expense. In [31] the CPU time required for closed form solution of the mass matrix, Eq. (6.8), is compared to the CPU time require for computing the mass matrix with different orders of Gauss quadrature. Significant savings in CPU times are indicated for the closed form integration, and this is of considerable importance in the development of efficient 3D compressible flow codes.

Of the three artificial dissipation models detailed in Chap. 2, the Lapidus dissipation model requires the least computations and minimal storage. The Lapidus dissipation Eq. (6.10) being highly nonlinear is evaluated using numerical integration. The order of the terms in Eq. (6.10) indicates the need for second order integration (in 3D, 8 integration points), but preliminary numerical experiments indicate that one-point Gauss quadrature appears adequate for most problems.

6.3 Computational Speed and Storage

The 3D vectorized finite element program was used to analyze flow-fields for problems ranging from a small number of nodes (a few thousands) to those needing large numbers of nodes (about 36,000). Experience gained from these problems indicate the speed of computations for this program to be 6×10^{-5} CPU seconds per time-step per node. This figure is competitive with existing finite difference codes. Another important concern is the storage required for the finite element program to run realistic problems. The VPS-32 has 32 million full precision (64 bits) words of central memory which translates to over 500 large pages (65,536 words make up a large page) of memory. The finite element program was written as an "in-core" program, which imposes a ceiling on the size of the model that can be used for flow simulation. The storage needed for a problem can be estimated from the relation,

$$MTOT = 40 \times NPOIN + 270 \times NELEM$$

where MTOT is the total storage required. NPOIN and NELEM are the number of nodes and elements, respectively. In terms of large pages MTOT/65,536 has the upper limit of 448, beyond which "page faulting" degrades the performance of the program. Thus, the maximum problem size that the finite element program can process is about 120,000 nodes. If larger problems are encountered major modifications to the program are required. A possible strategy would be to restructure the data in such a way that paging, even when it occurs, is handled

efficiently. Programming strategies such as "packing" and "unpacking" memory, recalculation instead of storage, and use of efficient input-output requests can also help in handling very large problems.

6.4 3D Sample Problems

The capability of the computational procedure to accurately calculate internal and external flowfields containing expansions, shock waves, and shear regions is demonstrated by two sample problems. Comparison solutions obtained by other methods are also presented to gauge solution accuracy and shock resolution.

6.4.1 Square Nozzle

The first problem presented is the flowfield in an expansion-recompression square nozzle (Fig. 6.3). The problem is reduced to one-fourth its size using symmetry with the flow region being bounded by two planes of symmetry, an upper wall and a side wall. The flow field is characterized by expansion waves emanating from the upper and the side walls in the region $0<x<5$. In the region $5<x<10$, the flow is recompressed resulting in shock waves emanating from the upper and side walls. The shock waves intersect at $x=17$ and reflect from the symmetry planes.

The inlet Mach number is 2.94 and a finite element mesh consisting of 7865 nodes and 6400 elements was used to march the solutions to steady-state. Convergence was indicated by a decrease in the L_2 norm of the density changes by three orders of magnitude. The computational procedure used a global time-step that satisfied the CFL criterion.

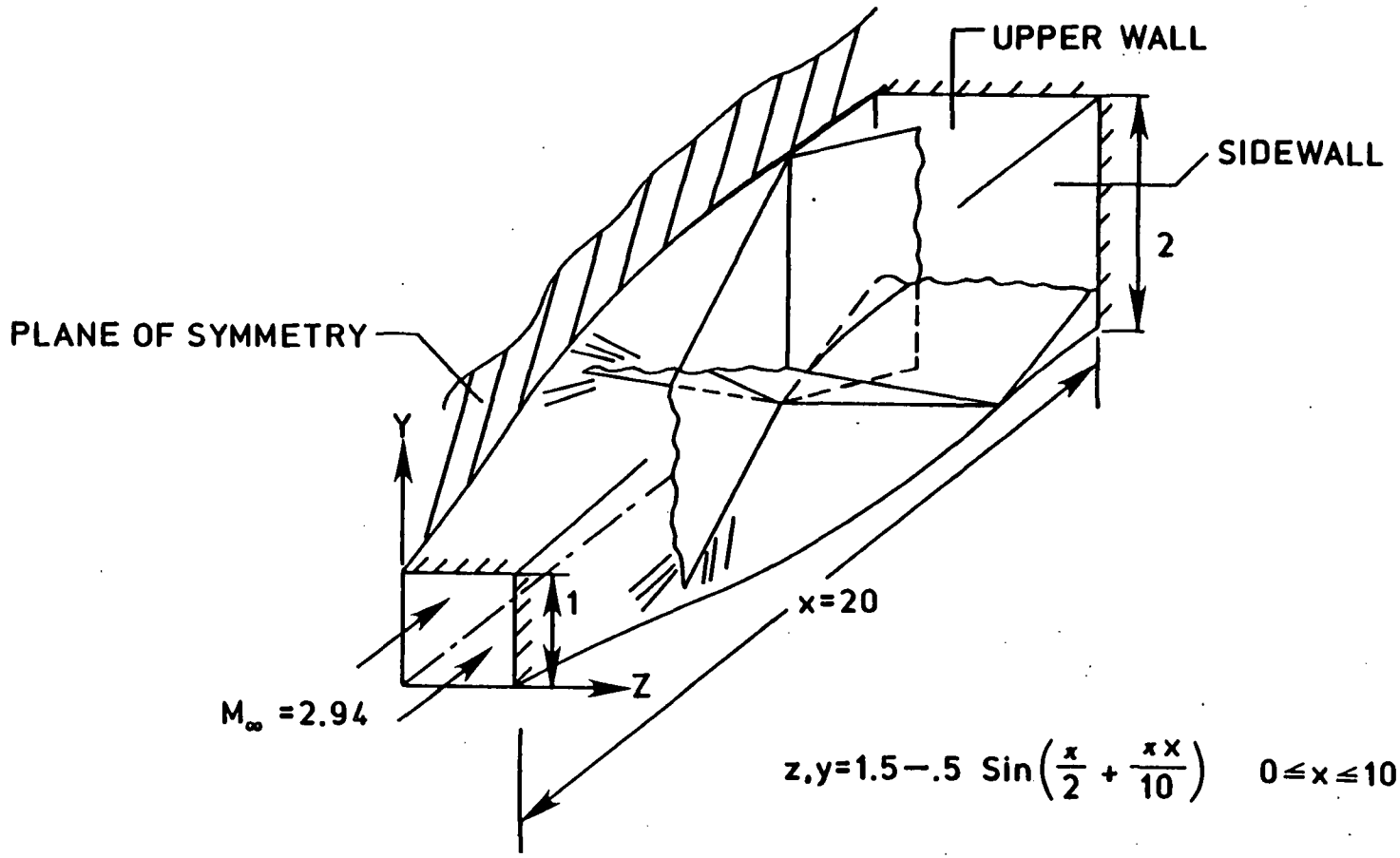


Fig. 6.3 Expansion-recompression square nozzle.

Figure 6.4 shows the pressure contours on the symmetry plane of the nozzle at z=0. The presence of the expansion waves at the inlet section and the formation of the shock wave and its subsequent reflection from the symmetry plane at x=17 is apparent in the figure. Figures 6.5 and 6.6 compare the finite element solution with the solutions obtained from a reference plane finite difference procedure [50]. Figure 6.5 shows the axial variation of pressure at the intersection of the planes of symmetry (y=z=0). The sharp increase in pressure at x=17 occurs due to the intersection of four shock waves emanating from the walls of the nozzle. The variation of pressure along the corner formed by the intersection of the upper wall and the side wall is shown in Fig. 6.6. These figures indicate that the flow features are accurately predicted by the finite element solution.

6.4.2 Scramjet Exhaust Flow

The second problem is the flow field associated with an outboard module of a hypersonic research aircraft. A hypersonic vehicle and typical flow features downstream of the nozzle exhaust are shown in Fig. 6.7. The external flow, both below and beside the nozzle, interacts with the nozzle outflow, and the complete geometric configuration to be modelled is detailed in Fig. 6.8.

To analyze the three dimensional shear flow, the problem was split up into three subproblems: (1) a 3D divergent nozzle, Fig 6.9a, (2) a 2D expansion over the vehicle body, Fig 6.9b, and (3) the 3D region downstream of the nozzle exit, Fig 6.9c.

The 3D flow field in the divergent nozzle was modelled with 8721 nodes and 7168 elements. The inlet Mach number is 1.657 and the

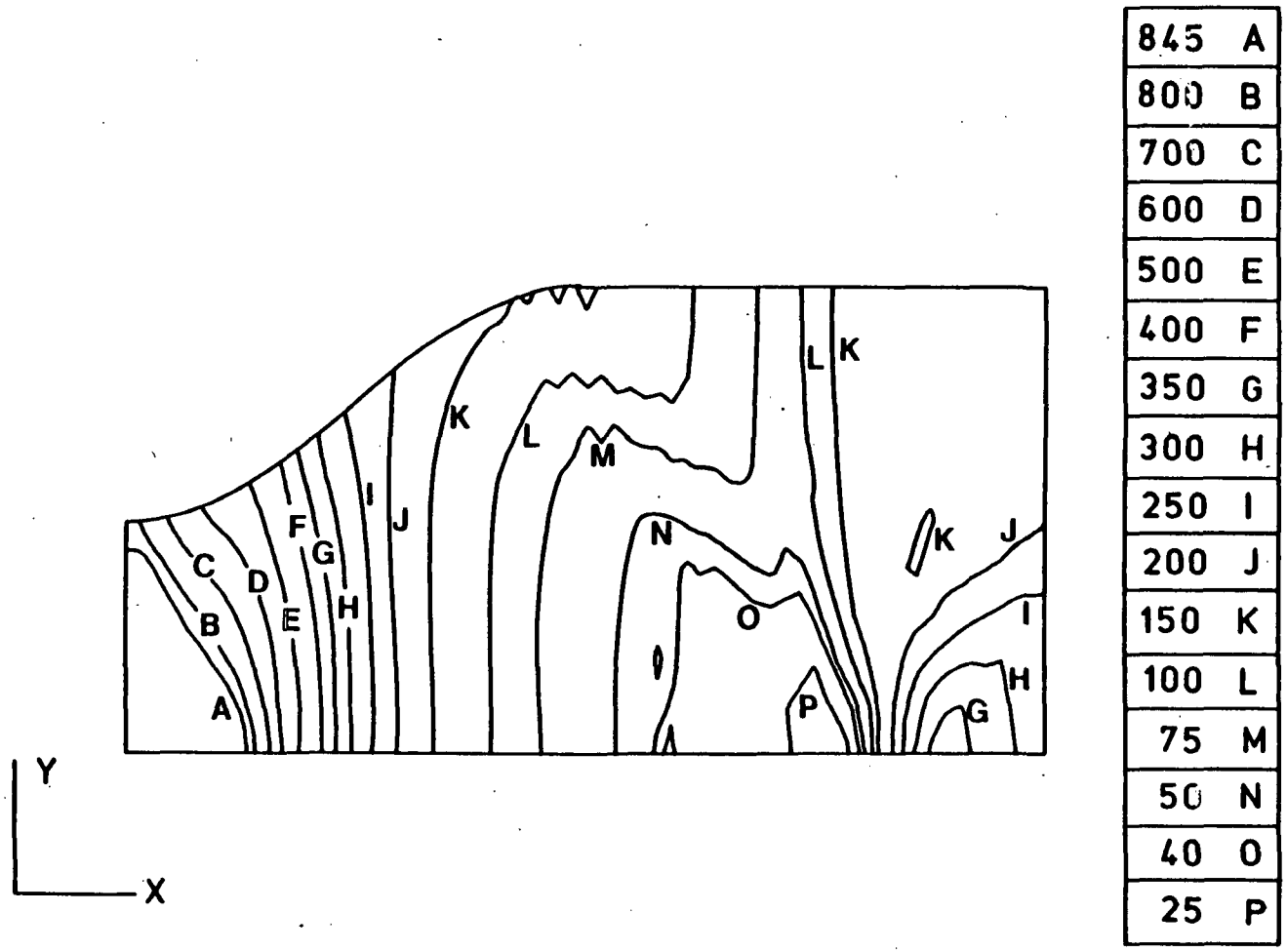


Fig. 6.4 Pressure contours at the symmetry plane (z=0) of nozzle.

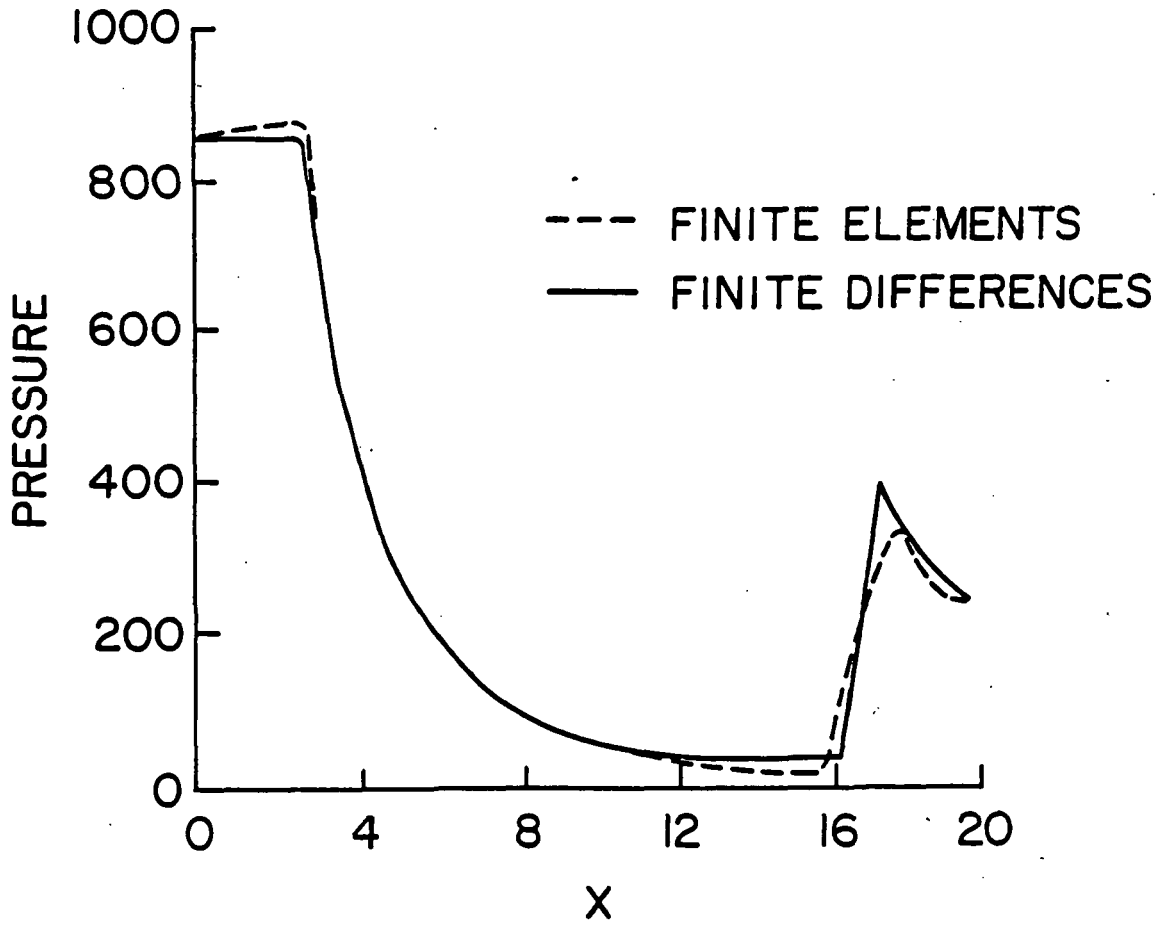


Fig. 6.5 Comparative pressure distributions along intersection of symmetry planes of square nozzle.

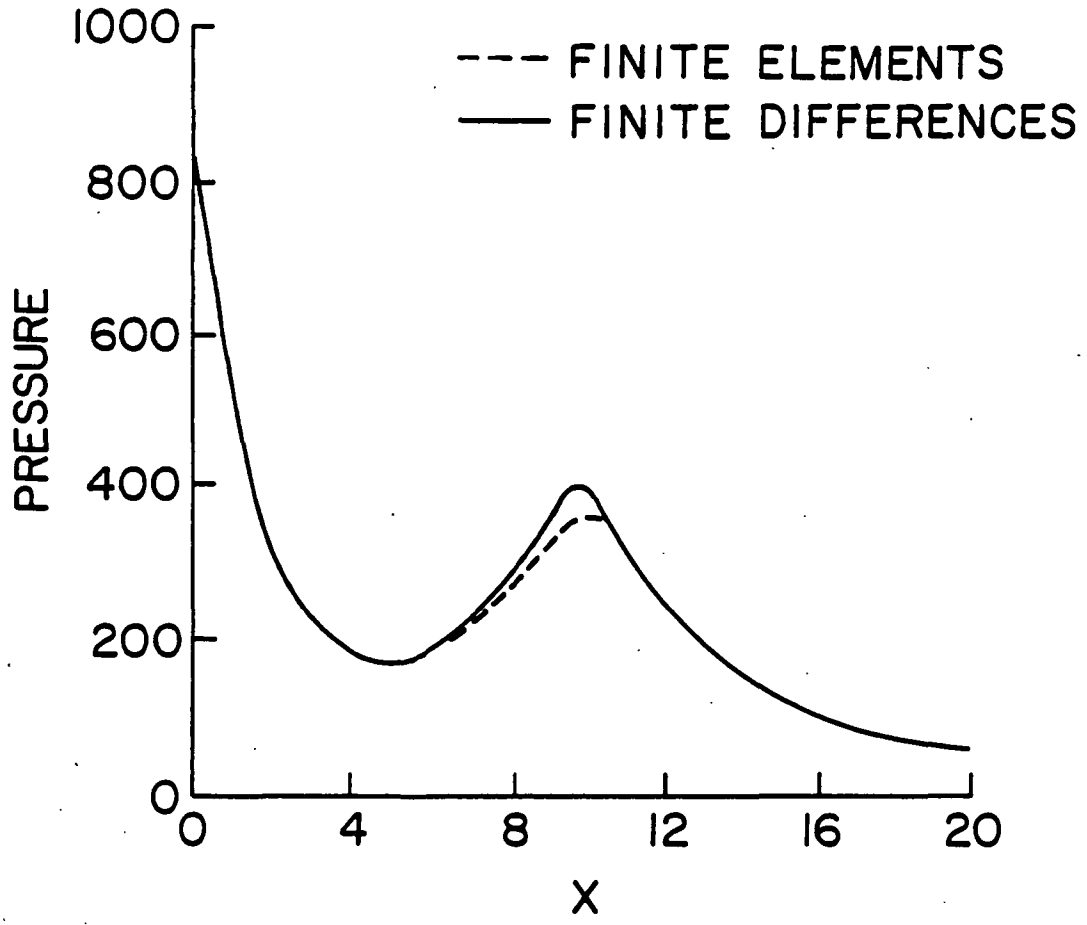


Fig. 6.6 Comparative pressure distributions along upperwall/sidewall corner of square nozzle.

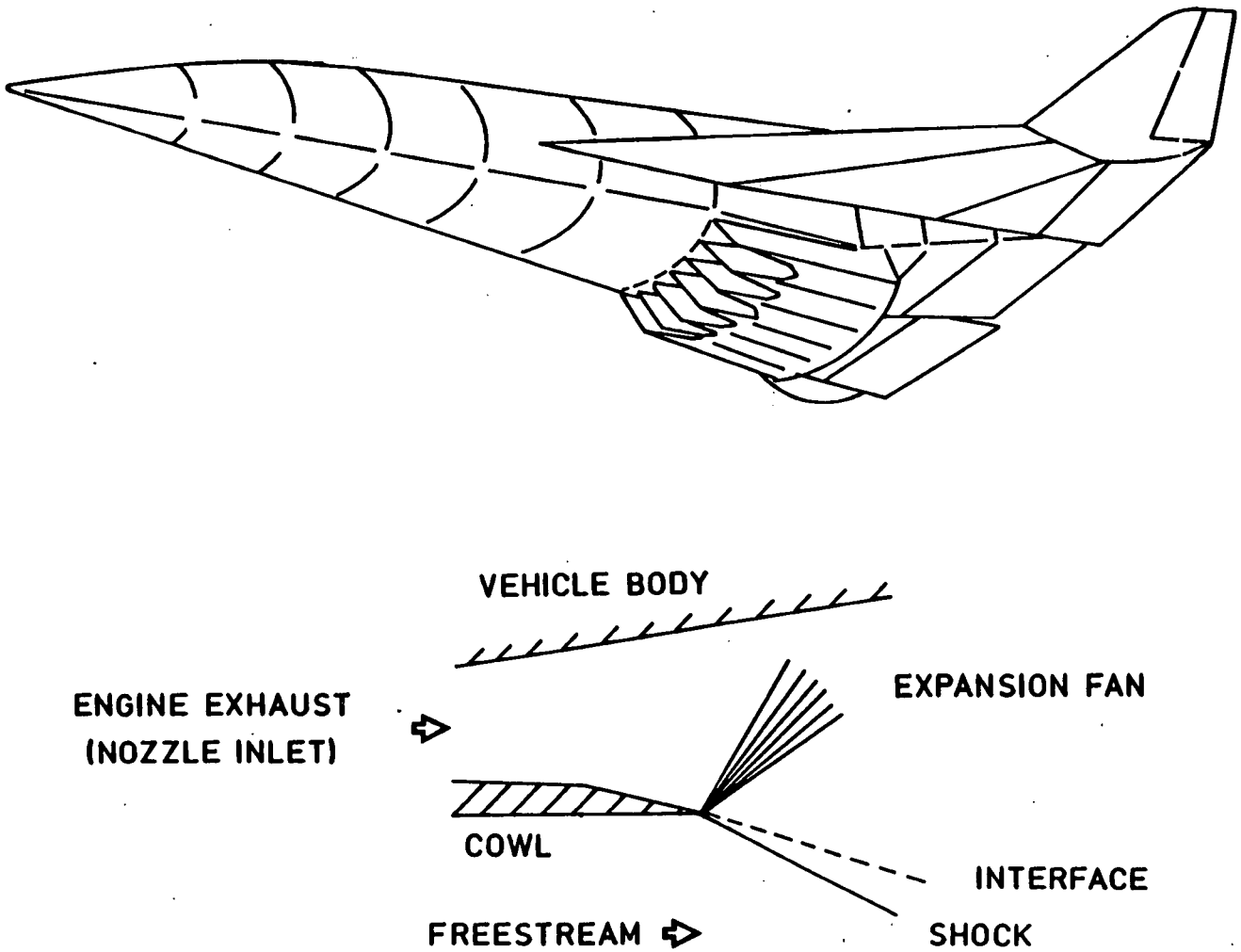


Fig. 6.7 Hypersonic research vehicle and flow features downstream of engine exhaust.

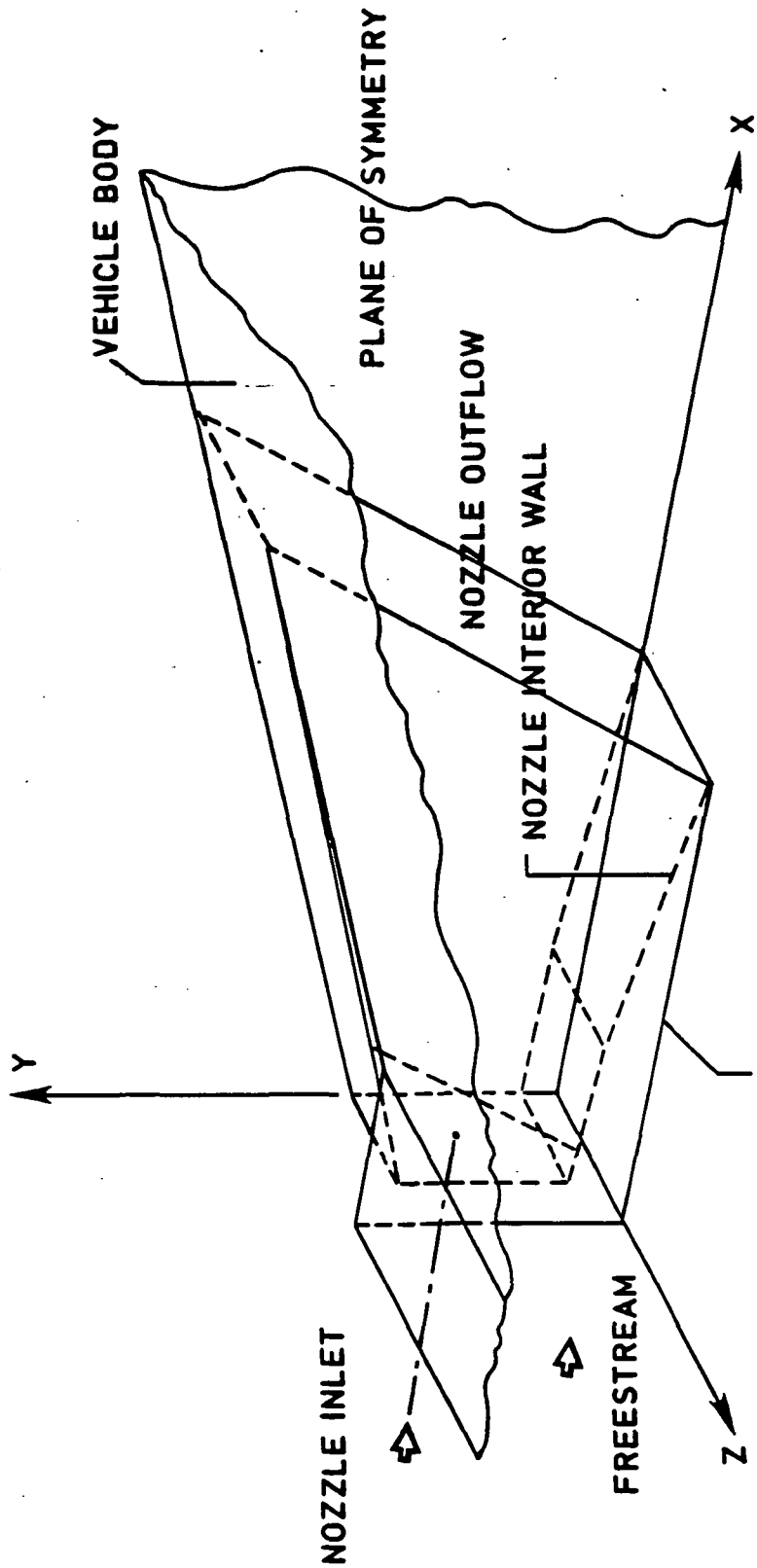
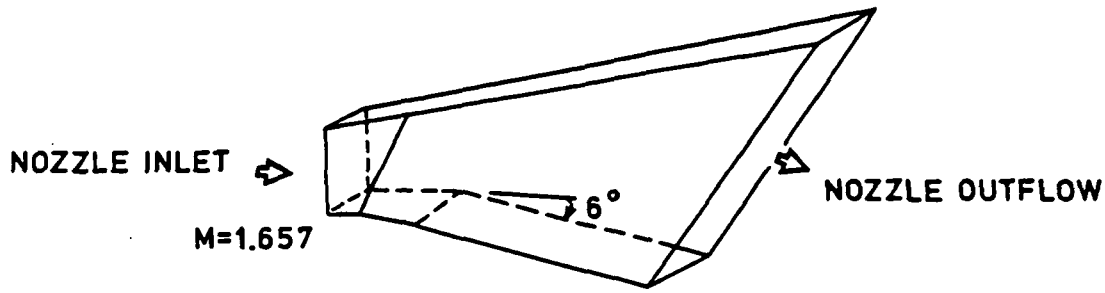
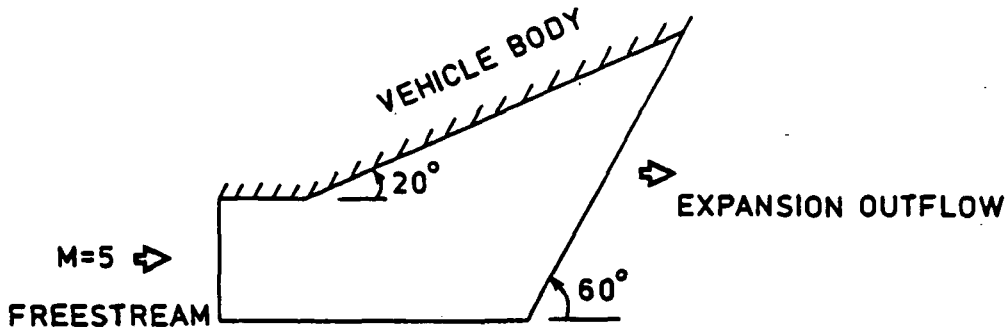


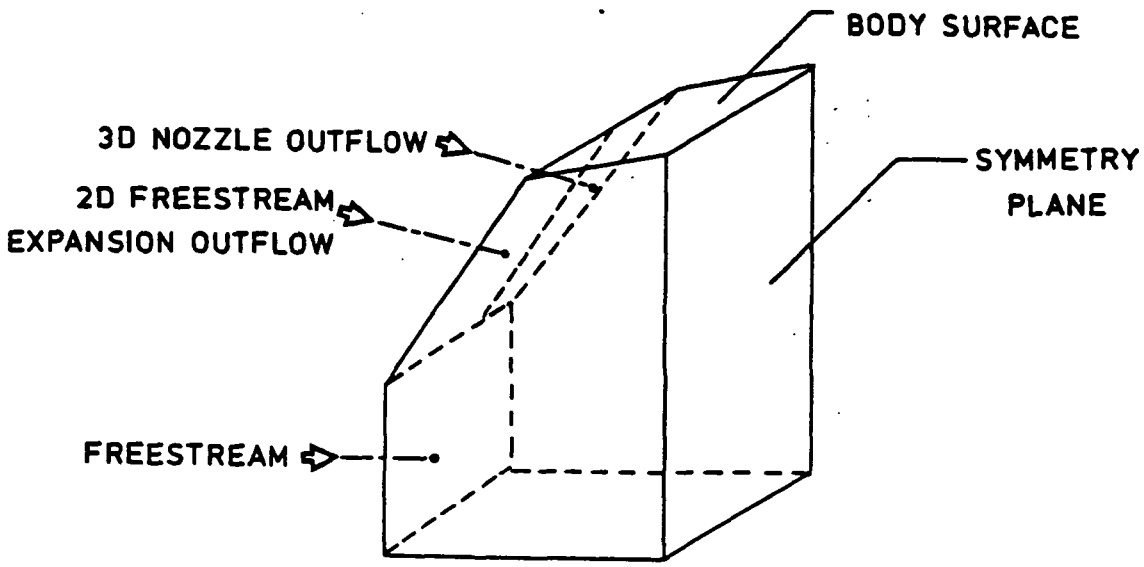
Fig. 6.8 Geometric configuration of an engine outboard module.



(a) 3D divergent nozzle



(b) 2D expansion over vehicle body



(c) 3D flow region downstream of nozzle.

Fig. 6.9 Subproblems for scramjet exhaust flow.

specific heat ratio is 1.27. Figure 6.10 shows the predicted pressure contours on a layer of elements on the symmetry plane of the nozzle. Flow enters parallel to the x axis, expands in the inlet section of the nozzle, and then a further expansion occurs downstream due to the downward turn of the nozzle walls. The second subproblem is a Prandtl-Meyer expansion with an inclined outflow plane. A 2D model consisting of 2057 nodes and 1920 quad elements was used to analyze this flowfield. The exit solutions from these two problems form the inlet condition for the third subproblem. This problem, the 3D shear region downstream of the nozzle, is modelled with a finite element mesh of 11,781 nodes and 10,240 elements.

The flowfield that results in the 3D shear region includes expansions, shocks and contact surfaces. The exhaust from the nozzle interacts with the freestream below the nozzle, producing an expansion region, a plume shock, and an interface or contact discontinuity. The flow field caused by the exhaust of the nozzle and the expansion over the vehicle body near the nozzle sidewall also results in a sidewall shock, interface and expansion waves. The intersection of these two families of expansions results in a complex three dimensional flowfield.

Figure 6.11 presents the predicted pressure contours on the outflow of the 3D shear region. The flow field on the left and the bottom of the outflow are seen to be relatively undisturbed. The 2D simple expansion waves are seen on the left by the horizontal contours H-N, and the lower region without contours remains undisturbed at free stream values. The presence of the shock envelope that comes off the bottom and the side walls of the nozzle is indicated by the closely

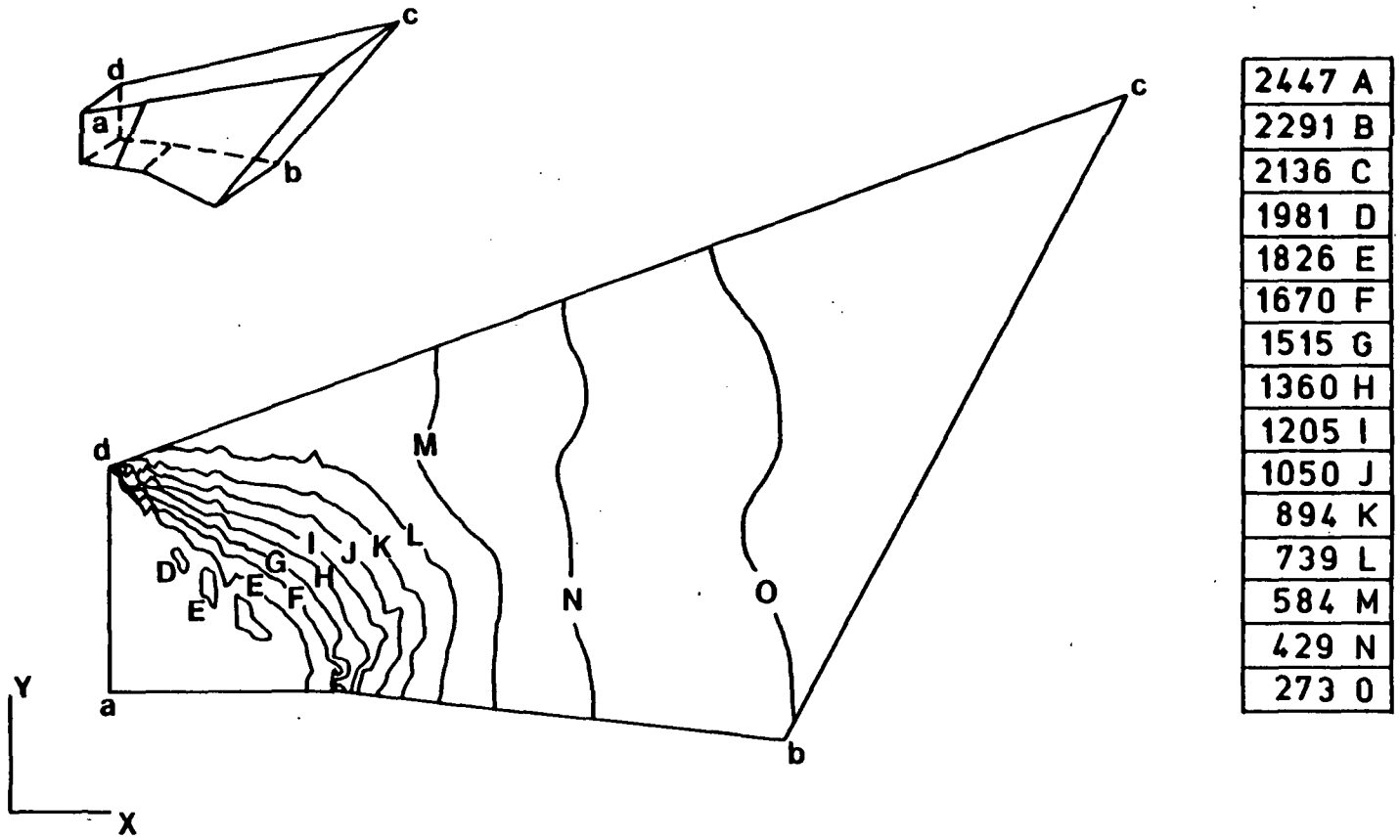


Fig. 6.10 Pressure contours on a layer of elements on symmetry plane of 3D divergent nozzle.

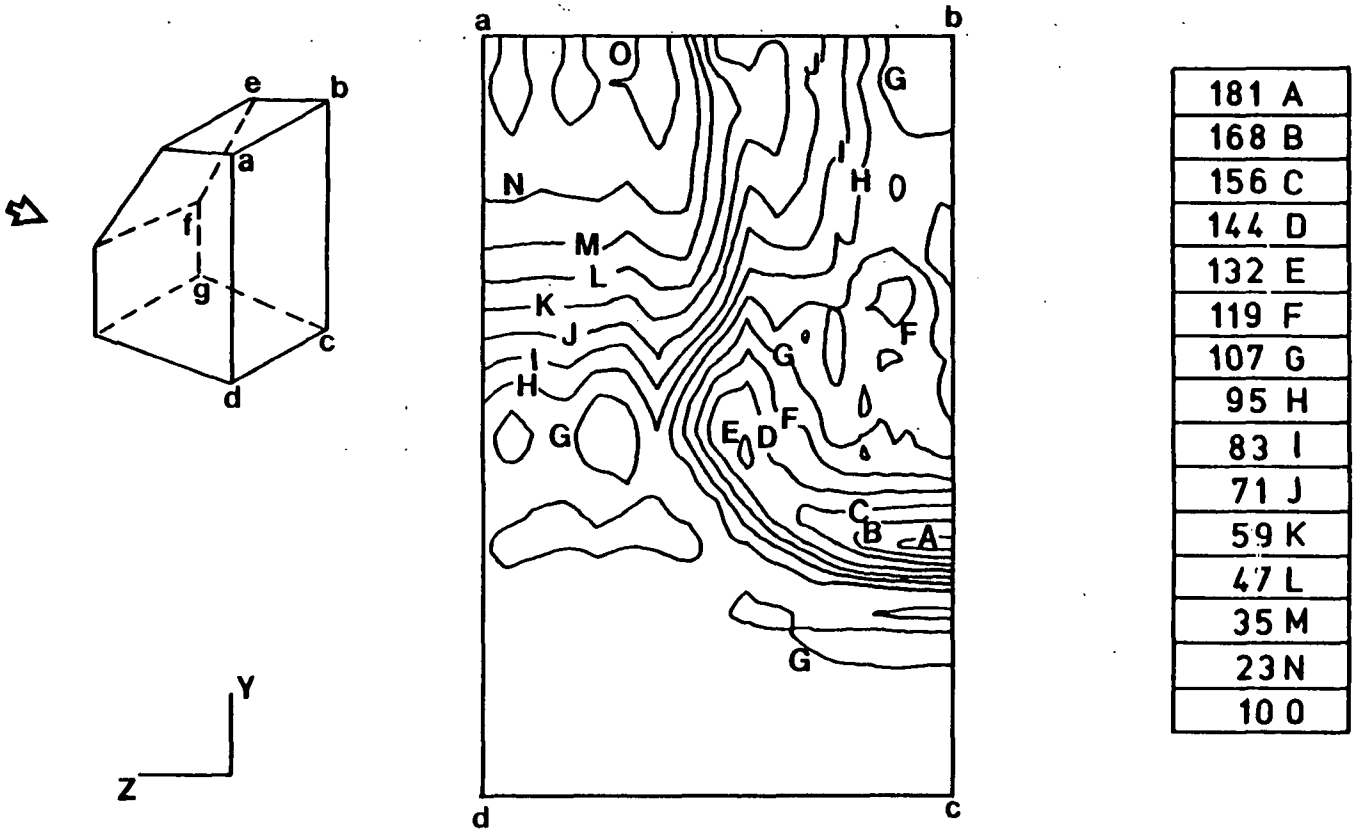


Fig. 6.11 Pressure contours on the outflow plane of the 3D shear region.

spaced contours while the complex expansion wave interactions appear in the upper right regions of the figure. The top right corner of the flow field is the region outside the intersection of the expansions and remains at freestream pressure.

The interaction between the nozzle outflow and the freestream is further illustrated by pressure contours on a layer of elements in the symmetry plane as shown in Fig. 6.12. The expansion waves generated and the shock that results (see Fig. 6.7) indicate the qualitative accuracy of the solution procedure.

The finite element results are compared with predictions from the GIM [44] program in Figs. 6.13 and 6.14. GIM (General Interpolants Method) combines features of the finite element and finite difference methods. Figure 6.13 shows the pressure distribution at the outflow plane of the shear region along a vertical line in the symmetry plane. The expansion region and the shock waves are displayed by both methods. The pressure distribution predicted by the finite element approach appears to be more realistic than the GIM results as indicated, for example, by the prediction of the sharper shock. Post-shock oscillations appears in the finite element results while the GIM results are seen to be overly smooth especially near the shock.

A further comparison of the two methods appears in Fig 6.14 where the pressure distribution at the outflow plane is plotted normal to the symmetry plane. The finite element scheme predicts the expansion near the symmetry plane and the weaker shock that results from the sidewall nozzle exhaust/freestream expansion. In contrast, the GIM results show an almost continuous expansion to freestream pressure values.

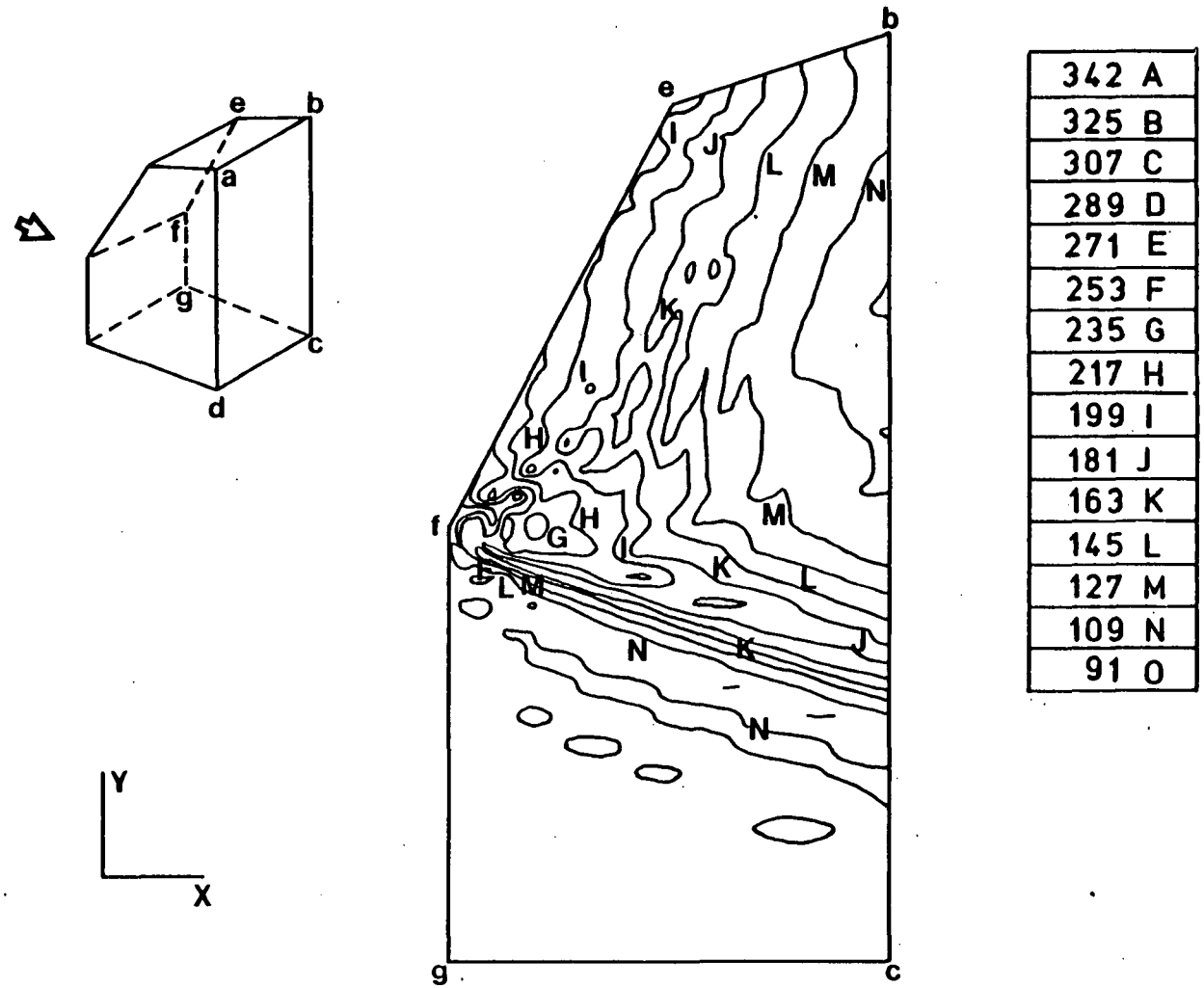


Fig. 6.12 Pressure contours on a layer of elements at symmetry plane of the 3D shear region.

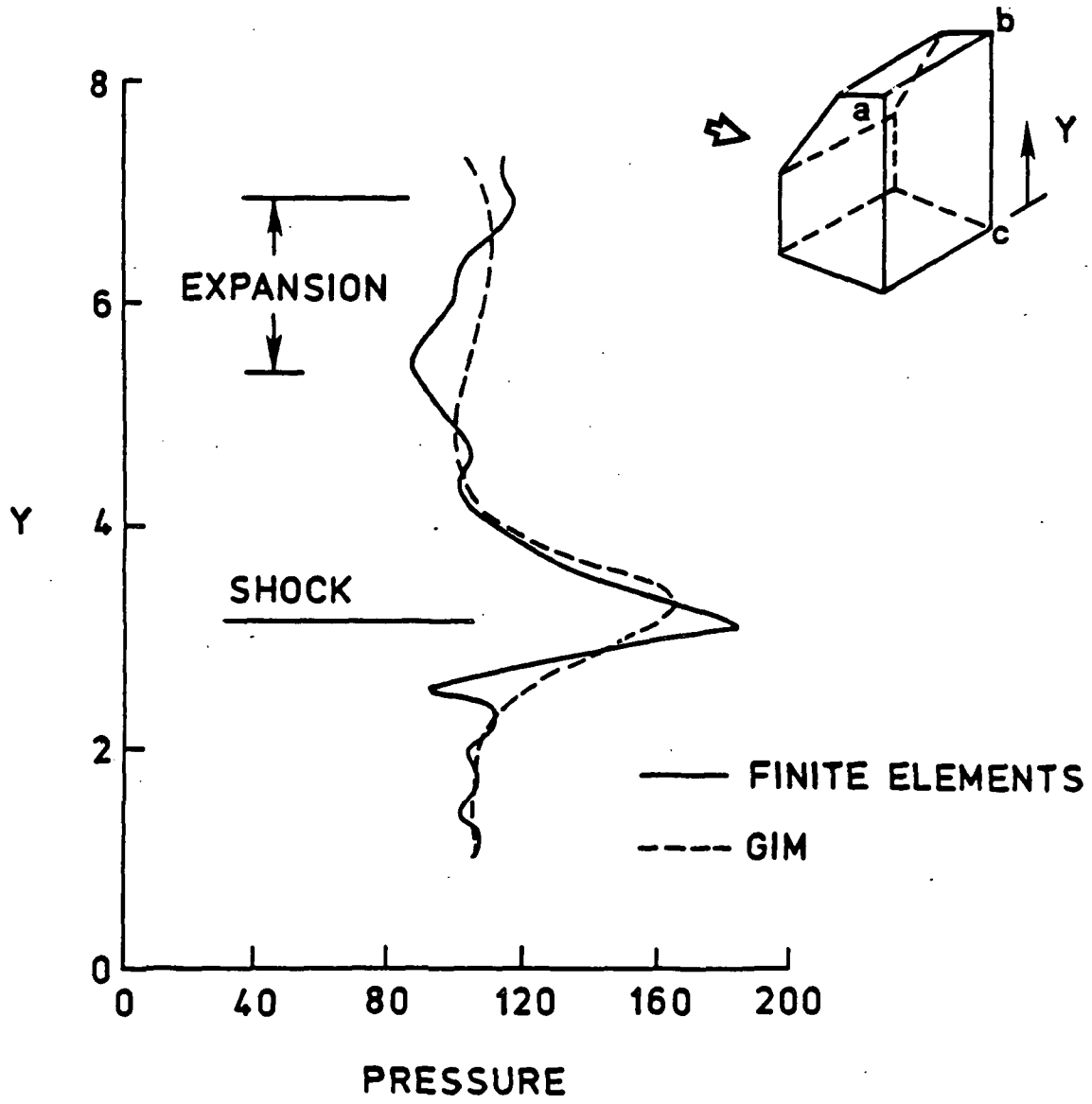


Fig. 6.13 Pressure distribution along vertical line in outflow plane of 3D shear region.

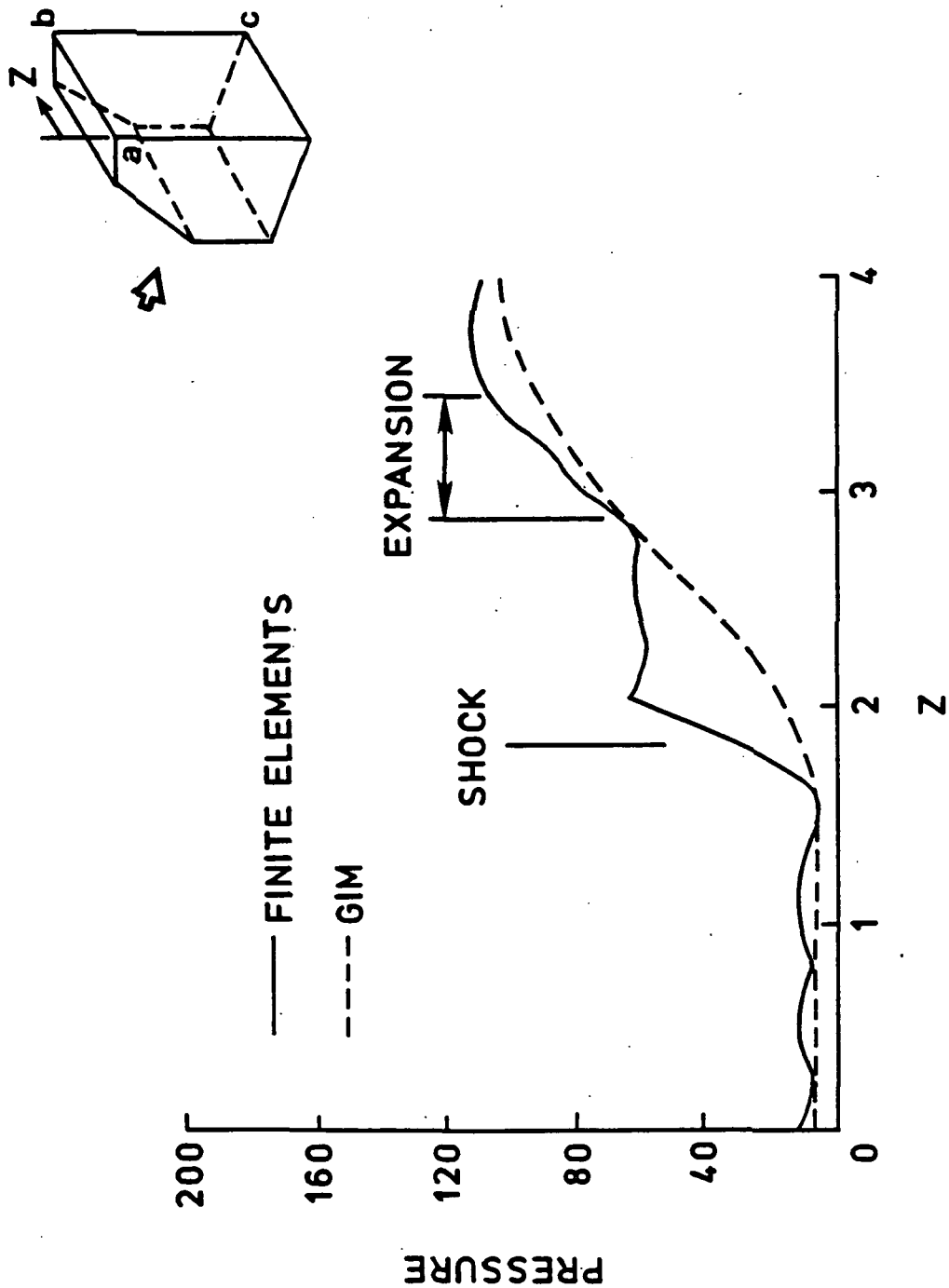


Fig. 6.14 Pressure distributions along horizontal line in the outflow plane of 3D shear region.

6.5 Closing Comments on the 3D Formulation

This chapter demonstrates the capability of the Taylor-Galerkin algorithm to simulate complicated inviscid flow for 3D problems. The computing speed of the program is competitive and indicates good possibilities for extension of the formulation to 3D viscous flows. The addition of the viscous terms appears to be straightforward and will be the subject of future research efforts.

Chapter 7

CONCLUDING REMARKS

7.1 Conclusions

The development of strategies for effective vectorization of finite element compressible flow programs is described in this study. The vectorization procedures are tailored to exploit the hardware and software characteristics of the NASA Langley VPS-32. The use of these strategies for 2D and 3D inviscid and viscous flow computations is demonstrated.

The basic principles of shock capturing methods are described. Two shock capturing finite element algorithms, the Taylor-Galerkin and the Petrov-Galerkin, are described. The Taylor-Galerkin algorithm uses explicit artificial dissipation, the Petrov-Galerkin algorithm is based on streamline upwind methodology. The use of three explicit dissipation models for the Taylor-Galerkin algorithm is described. Results obtained using Lapidus, MacCormack-Baldwin, and Jameson dissipation models are compared.

The Taylor-Galerkin algorithm with MacCormack-Baldwin dissipation and the Petrov-Galerkin algorithm are used to solve a variety of 2D inviscid flow problems. Comparisons for the two algorithms are made using criteria such as solution quality, shock resolution, computational speed and storage requirements. Results obtained show good

shock capturing properties for both methods. The Taylor-Galerkin algorithm exhibits local spurious oscillations at compression and expansion corners. The Petrov-Galerkin formulation shows minimal oscillations and good shock resolution. Results for the hypersonic flow over a blunt leading edge using the Petrov-Galerkin algorithm indicates the need to implement an "integration by parts" procedure to ensure flux conservation.

The computational speeds for the formulations indicate the Taylor-Galerkin algorithm to be about three times faster than the Petrov-Galerkin algorithm. Local time-stepping procedures implemented on both formulations show the capability of the Taylor-Galerkin algorithm to be run at higher Courant numbers, around 0.6, compared to about 0.3 for the Petrov-Galerkin algorithm. A comparison of the storage needed for the 2D programs indicates that Petrov-Galerkin algorithm needs only 90% of the storage needed by the Taylor-Galerkin algorithm. The size of storage needed may not be critical for 2D problems but becomes significant for 3D computations.

The vectorization strategies developed for the 2D inviscid Petrov-Galerkin algorithm is extended to include the effects of viscosity and heat conduction. The 2D viscous Petrov-Galerkin algorithm is validated by simulating the supersonic flow over a flat plate. Results obtained from the Petrov-Galerkin algorithm are compared with results from a finite difference method and a 2D viscous Taylor-Galerkin algorithm. The comparison of computational speeds of the two viscous algorithms indicates the Taylor-Galerkin algorithm to be twice as fast as the Petrov-Galerkin algorithm. The slower speeds for the Petrov-Galerkin for both the inviscid and viscous algorithms are due to

the need for numerical integral evaluations. Comparisons of storage needed for the two viscous algorithms shows that the Petrov-Galerkin algorithm needs less than 50% of the storage needed by the Taylor-Galerkin algorithm

The extension of the vectorization strategies for 3D inviscid Taylor-Galerkin computations is described. Two sample problems are shown to demonstrate the capability of the finite element procedure to model flow details such as shocks, expansions and shear layers for complicated three dimensional compressible flows. Results obtained using the finite element procedure are compared with results from the reference plane finite difference method and the General Interpolants Method, GIM. The computational speeds obtained with the 3D vectorized Taylor-Galerkin procedure is seen to be competitive with existing finite difference codes.

The development of an efficient vectorization procedure for general explicit finite element flow algorithms and applications of this procedure to 2D and 3D inviscid and viscous flows are demonstrated. The computational rates obtained for both finite element algorithms indicate the ability of the procedure to handle complex three dimensional compressible flows requiring nodes that number in the hundreds of thousands. Much research remains to be done to be able to accurately predict aerothermal-structural interactions for high speed vehicles, but a computational procedure has been developed in this study that will play a major role in achieving this goal.

7.2 Recommendations for Further Research

The finite element methods described in this study possess good shock capturing properties. Shock resolution can be enhanced further by local mesh refinements. In most flow situations the exact location of shocks or other gradients is not known a-priori. This underlines the need to develop adaptive mesh refinement procedures. Refinement procedures can be developed independent of the finite element algorithm and can be interfaced with either formulation.

The analysis of high speed viscous flow requires very refined mesh spacings close to the surface of the body to capture details of the boundary layer. The computational expense for viscous analyses are usually more expensive than inviscid analyses. A good procedure for viscous analyses is to start off with an inviscid analysis of the flow region. The results obtained from the inviscid analysis contain details on features such as shocks and expansions. The development of programs to accurately interpolate the mesh information from inviscid to viscous grids is very desirable since it results in better initial conditions and faster rates of convergence for the viscous problem.

The Petrov-Galerkin formulation has demonstrated properties for good shock resolution, solution accuracy, fast convergence and lesser storage requirements. The main drawback of this method is the slower computational speeds for both inviscid and viscous formulations. The development of accurate one point integration procedures will improve processing rates and result in a finite element formulation with excellent computational properties.

The development of accurate one point integration schemes will also help in improving the processing rates of the Taylor-Galerkin algorithm. The timing data for the Taylor-Galerkin formulation indicates that over 50% of processing time is spent in the numerical integration needed for addition of artificial viscosity. Use of accurate one-point integration schemes could double the processing rates for the Taylor-Galerkin algorithm.

Viscous flow problems require very small grids spacings at the walls and these spatial dimensions may necessitate implicit treatment of the elements that lie inside the boundary layer. Vectorization procedure may have to be developed for mixed implicit-explicit solution procedures.

The current interest in hypersonic flows in the range of Mach 10 to Mach 25 indicates the need to include real gas effects in the solution procedure. The need to develop vectorization strategies for these effects is essential to solve realistic hypersonic viscous flow problems.

REFERENCES

1 Holt, M., Numerical Methods in Fluid Dynamics, Springer Series in Computational Physics, Springer-Verlag, New York, 1977.

2 Huebner, K. H. and Thornton, E. A., The Finite Element Method for Engineers, Second Edition, John Wiley, NY, 1982.

3 Zienkiewicz, O. C. in collaboration with Lohner, R., Morgan, K., and Nakazawa, S., "Finite Elements in Fluid Mechanics - A Decade of Progress," Chapter to appear in Finite Elements in Fluids, Vol. 5, John Wiley and Sons.

4 Bey, K. S., "An Evaluation of a Taylor-Galerkin Algorithm for High Speed Inviscid Flows Using Quadrilateral and Triangular Elements," Masters Thesis, Old Dominion University, May 1986.

5 Lohner, R., "Finite Element Methods for Hyperbolic Partial Differential Equations," Ph.D. Dissertation, University College of Swansea, October 1984.

6 Ecer, A. and Akay, H. U., "Investigation of Transonic Flow in a Cascade using the Finite Element Method," AIAA Journal, Vol. 19, No. 9, (1981), pp.1174-1182.

7 Ecer, A. and Akay, H. U., "A Finite Element Formulation of the Euler Equations for the Solution of Steady Transonic Flows," AIAA 20th Aerospace Sciences Meeting, Orlando, Florida, Jan. 11-14, 1982, AIAA 82-0062.

8 Argrand, F., Dervieux, A., Boulard, V., Periaux, J., and Vijayasundaram, G., "Transonic Euler Simulations by Means of Finite Element Explicit Schemes," AIAA 6th Computational Fluid Dynamics Conference, Danvers, MA, July 13-15, 1983, AIAA 83-1924.

9 Argrand, F., Boulard, V., Periaux, J., Dervieux, A., and Vijayasundaram, G., "Triangular Finite Element Methods for the Euler Equations," Proceedings of the 6th International Symposium on Computing Methods in Applied Sciences and Engineering, Versailles, France, Dec. 12-16, 1983.

10 Jameson, A., Baker, T. J., and Weatherhill, N. P., "Calculation of Inviscid Transonic Flow Over a Complete Aircraft," AIAA 24th Aerospace Sciences Meeting, Reno, NV, Jan. 6-9, 1986, AIAA 86-0103.

11 Lohner, R., Morgan, K., and Zienkiewicz, O. C., "The Solution of Nonlinear Hyperbolic Equation Systems by the Finite Element Method," International Journal of Numerical Methods in Fluids, Vol. 4, (1984) pp. 1043-1063.

12 Lohner, R., Morgan, K., and Zienkiewicz, O.C., "The Use of Domain Splitting with an Explicit Hyperbolic Solver," Computer Methods in Applied Mechanics and Engineering, Vol. 45, (1984), pp. 313-329.

13 Zienkiewicz, O. C., Lohner, R., and Morgan, K., "High Speed Inviscid Compressible Flows by the Finite Element Method," Presented at MAFELAP-IV Conference, May 1-4, 1984, Brunel, U.K.

14 Hughes, T. J. R., Franca, L. P., and Mallet, M., "A New Finite Element Method for Computational Fluid Dynamics, I. Symmetric Forms of the Compressible Euler and Navier-Stokes Equations and the Second Law of Thermodynamics," to appear in Computer Methods in Applied Mechanics and Engineering, 1986.

15 Hughes, T. J. R., Mallet, M., and Mizukami, A., "A New Finite Element Method for Computational Fluid Dynamics: II. Beyond SUPG," to appear in Computer Methods in Applied Mechanics and Engineering, 1986.

16 Hughes, T. J. R, Mallet, M., and Franca, L. P., "Entropy Stable Finite Element Methods for Compressible Fluids: Application to High Mach Number Flows with Shocks," to appear in Finite Element Methods for Nonlinear Problems, (ed. P. Bergan), Springer-Verlag.

17 Lohner, R., Morgan, K., Vahdati, M., Boris, J. P., and Book, D. L., "FEM-FCT Combining Unstructured Grids with High Resolution," C/R/539/86, University College of Swansea, Wales, U.K., Jan. 1986.

18 Mallet, M., Private Communications.

19 Dechaumphai, P. and Thornton, E. A., "The Role of PATRAN in Finite Element Computational Fluid Dynamics," 1986 PATRAN User's Conference, Newport Beach, California, June 3-5, 1986.

20 Moretti, G. and Bleich, G., "3D Inviscid Flow about Supersonic Blunt Cones at Angle of Attack," Sandia Laboratories Report, SC-68-3728, Albuquerque, NM, 1968.

21 Lax, P. D., "Weak Solutions of Nonlinear Hyperbolic Equations and their Numerical Computations," Communications in Pure and Applied Mathematics, Vol. VII, pp. 159-193, 1954.

22 von Neumann, J. and Richtmyer, R. D., "Method for the Numerical Calculation of Hydrodynamic Shocks," Journal of Applied Physics, Vol. 21, (1950), pp. 232-237.

23 MacCormack, R. W., "The Effect of Viscosity in Hypervelocity Impact Cratering," AIAA 69-354, Cincinnatti, OH, 1969.

24 Beam, R. M. and Warming, R. F., "An Implicit Finite Difference Algorithm for Hyperbolic Systems in Conservation Law Form," Journal of Computational Physics, Vol. 22, (1976), pp. 87-110.

25 Burstein, S. and Mirin, A. A., "Third Order Difference Methods for Hyperboli Equations," Journal of Computational Physics, Vol. 5, (1970), pp. 547-571.

26 Steger, J. and Warming, R. F., "Flux Vector Splitting of Inviscid Gasdynamic Equations with Applications to the Finite Difference Methods," NASA TM-78605, July, 1979.

27 Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors and Difference Schemes," Journal of Computational Physics, Vol. 43, pp. 357-372, 1981.

28 van Leer, B., "Toward the Ultimate Conservative Difference Scheme: V. A Second Order Sequel to Gudunovs Method," Journal of Computational Physics, Vol. 32, (1979), pp. 101-136.

29 Pulliam, T. H., "Artificial Dissipation Models for the Euler Equations," AIAA 85-0438, Reno, NV, Jan. 1985.

30 Donea, J., "A Taylor-Galerkin Method for Convective Transport Problems," Numerical Methods in Thermal Problems, edited by R. W. Lewis, J. A. Johnson and W. R. Smith, Proceedings of the Third International Conference, Seattle, WA, August 2-5, 1983, Pineridge Press, Swansea, U. K.

31 Bey, K. S., Thornton, E. A., Dechaumphai, E. A., and Ramakrishnan, R., "A New Finite Element Approach for Prediction of Aerothermal Loads-Progress in Inviscid Flow Computations," AIAA 7th Computational Fluid Dynamics Conference, Cincinnatti, OH, July 15-17, 1985, AIAA 85-1533-CP.

32 Lapidus, A., "A Detached Shock Calculation by Second Order Finite Differences," Journal of Computational Physics, Vol. 2 (1967), pp. 154-177.

33 MacCormack, R. W. and Baldwin, B. S., "A Numerical Method for Solving the Navier-Stokes Equations with Application to Shock Boundary Interactions," AIAA 75-1, 1975.

34 Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solution of the Euler Equations by Finite Volume Method Using Runge Kutta Time Stepping Schemes," AIAA 81-1259, June 1981.

35 Hughes, T. J. R. and Brooks, A. N., "A Multidimensional Upwind Scheme with no Crosswind Diffusion," Finite Element Methods for Convection Dominated Flows (ed. T. J. R. Hughes), ASME, New York, 1979, pp.19-35.

36 Hughes, T. J. R. and Brooks, A. N., "A Theoretical Framework for Petrov-Galerkin Methods with Discontinuous Weighting Functions: Application to the Streamline Upwind Procedure," Finite Elements in Fluids, Vol. IV (eds. R. H. Gallagher et al.), John Wiley and Sons, London.

37 Hughes, T. J. R. and Tezduyar, T. E., "Finite Element Methods for First Order Hyperbolic Systems with Particular Emphasis on the Compressible Euler Equations," Computer Methods in Applied Mechanics and Engineering, Vol. 45, pp. 217-284.

38 Harten, A., "On the Symmetric Form of the System of Conservation Laws with Entropy," Journal of Computational Physics, Vol. 49, pp. 151-164.

39 Mallet, M., "A Finite Element Method for Computational Fluid Dynamics," Ph.D Dissertation, Stanford University, Stanford, California, 1985.

40 Noor, A. K. and Lambiotte, J. J., "Finite Element Dynamic Analysis on the CDC-STAR 100 Computer," Journal of Computers and Structures, Vol. 10, 1979, pp. 7-19.

41 Zienkiewicz, O. C., The Finite Element Method, 3rd edition, McGraw-Hill Book Company, New York, 1977.

42 Lohner, R., Morgan, K., and Peraire, J., "A Simple Extension to Multidimensional Problems of the Artificial Viscosity Model due to Lapidus," to appear in Communications in Applied Numerical Methods, 1986.

43 Anderson, D. A., Tannehill, J. C., and Pletcher, R. H., Computational Fluid Mechanics and Heat Transfer, Hemisphere Publishing Corporation, New York, 1984.

44 Spradley, L. W., Anderson, P. G., and Pearson, M. L., "Computation of Three Dimensional Nozzle Exhaust Flow Fields with the GIM Code," NASA CR-3042, 1978.

45 Billig, F. S., "Shock-Wave Shapes around Spherical- and Cylindrical-Nosed Bodies," Journal of Spacecraft and Rockets, Vol. 4, No. 6, June, 1967.

46 Walters, R. W. and Dwoyer, D. L., "An Efficient Iteration Strategy Based on Upwind Relaxation Schemes for the Euler Equations," AIAA 85-1529-CP, presented at the AIAA 7th CFD Conference, Cincinnatti, OH, July 15-17, 1985.

47 Lyubimov, A. N. and Rusanov, V. V., "Gas Flows Past Blunt Bodies: Part II. Tables of the Gasdynamic Functions," NASA TT F-715, 1973.

48 Le Balleur, J. C., Viviand, H., and Peyret, R., "Numerical Studies in High Reynolds Number Aerodynamics," Symposium on Computers in Aerodynamics, Polytechnic Institute of New York, Farmingdale, NY, 1979.

49 Carter, J. E., "Numerical Solutions of the Navier-Stokes Equations for the Supersonic Laminar Flow over a Two-Dimensional Compression Corner," NASA TR-385 July 1972.

50 Thornton, E. A., Dechaumphai, P., and Vemaganti, G., "A Finite Element Approach for Prediction of Aerothermal Loads," AIAA/ASME 4th Joint Fluid Mechanics, Plasma Dynamics and Lasers Conference, May 12-14, Atlanta, Georgia, 1986, AIAA 86-1050-CP.

51 Dash, S. and DelGuidice, P., "Shock Capturing Finite Difference and Characteristic Reference Plane Techniques for the Prediction of Three Dimensional Nozzle Flowfields," NASA CR-145366, 1978.

APPENDIX A

COMPUTATION OF NODAL SECOND DERIVATIVES

The dissipation models of MacCormack-Baldwin, Eq. (2.34) and Jameson, Eq. (2.40) need second and third derivatives of nodal quantities such as pressure and the conservation variables. The second derivatives at the nodes of an element can be obtained variationally from the first derivative defined within that element using Greens formula. For example, the second derivatives in the x direction can be written as,

$$U_{,xx} = (U_{,x})_x \tag{A.1}$$

Let

$$L(u) = u_{,x} \tag{A.2}$$

The adjoint operator for L(u) is given by,

$$L^*(v) = -v_{,x} \tag{A.3}$$

By Greens formula,

$$\int_A [v\, L(u) - u\, L^*(v)]\, dA = \int_D [Qi - Pj] \cdot \hat{n}\, ds \tag{A.4}$$

where A is the region enclosed by D, the boundary of the region. $\hat{n}$ is the outward normal to the boundary D. P and Q are evaluated on the boundary and depend on the differential equation. The values of P and Q are given by,

$$\begin{aligned} P &= 0 \\ Q &= uv \end{aligned} \tag{A.5}$$

Substituting the above in Eq. (A.4) results in,

$$\int_A [v\ L(u) - u\ L^*(v)]\ dA = \int_A uv\ (i \cdot n)\ ds \tag{A.6}$$

The boundary term is assumed to vanish for outflow surfaces and Eq. (A.6) reduces to,

$$\int_A vu_{,x}\ dA = -\int_A u\ v_{,x}\ dA \tag{A.7}$$

where u and v are functions of x and are well behaved. Let u and v be defined as,

$$\begin{aligned} v &= N \\ u &= U_{,x} \end{aligned} \tag{A.8}$$

where N is a matrix of element interpolation functions and U is a nodal variable. Substituting the values of u and v in Eq. (A.7) results in,

$$\int_A N\ U_{,xx}\ dA = -\int_A U_{,x}\ N_{,x}\ dA \tag{A.9}$$

Assume an interpolation of the second derivatives at the nodes given by,

$$U_{,xx} = N\ \{U_{,xx}\} \tag{A.10}$$

and the first derivatives is evaluated inside the element, typically at a Gauss point. The resulting equation is,

$$\int_A \{N\}\ [N]\ dA\ \{U_{,xx}\} = -\int_A N_{,x}\ dA\ U_{,x} \tag{A.11}$$

which can be written following Eq. (2.20) as,

$$[M]\ \{U_{,xx}\} = -\int_A N_{,x}\ dA\ U_{,x} \tag{A.12}$$

An alternate procedure to obtain the nodal second derivatives is to assume a linear variation of the first derivatives at the nodes,

$$U_{,x} = N \{U_{,x}\} \tag{A.13}$$

The second derivatives at the nodes can then be written as,

$$[M] \{U_{,xx}\} = \int_A N_{,x} N \, dA \{U_{,x}\} \tag{A.14}$$

The procedure adopted in this dissertation is to compute the second derivatives of pressure as given by Eq. (A.12) but to compute the second and third derivatives of the conservation variables along the lines of Eq. (A.14).

APPENDIX B

DEFINITION OF JACOBIAN MATRICES FOR COMPRESSIBLE FLOW EQUATIONS

The following matrices are the Jacobians of the Euler fluxes with respect to the conservation variables:

$$A_1 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ a_{21} & -(\gamma-3)u_1 & -(\gamma-1)u_2 & -(\gamma-1)u_3 & (\gamma-1) \\ -u_1u_2 & u_2 & u_1 & 0 & 0 \\ -u_1u_3 & u_3 & 0 & u_1 & 0 \\ a_{51} & a_{52} & -(\gamma-1)u_1u_2 & -(\gamma-1)u_1u_3 & \gamma u_1 \end{pmatrix}$$

$$a_{21} = (\frac{\gamma-1}{2})u^2 - u_1^2$$

$$a_{51} = u_1\Big((\gamma-1)u^2 - \gamma e\Big)$$

$$a_{52} = \gamma e - (\gamma-1)(u_1^2 + u^2/2)$$

$$A_2 = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ -u_1u_2 & u_2 & u_1 & 0 & 0 \\ a_{31} & -(\gamma-1)u_1 & -(\gamma-3)u_2 & -(\gamma-1)u_3 & (\gamma-1) \\ -u_2u_3 & 0 & u_3 & u_2 & 0 \\ a_{51} & -(\gamma-1)u_1u_2 & a_{53} & -(\gamma-1)u_2u_3 & \gamma u_2 \end{pmatrix}$$

$$a_{31} = (\frac{\gamma-1}{2})u^2 - u_2^2$$

$$a_{51} = u_2\Big((\gamma-1)u^2 - \gamma e\Big)$$

$$a_{53} = \gamma e - (\gamma-1)(u_2^2 + u^2/2)$$

$$A_3 = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ -u_1u_3 & u_3 & 0 & u_1 & 0 \\ -u_2u_3 & 0 & u_3 & u_2 & 0 \\ a_{41} & -(\gamma-1)u_1 & -(\gamma-1)u_2 & -(\gamma-3)u_3 & (\gamma-1) \\ a_{51} & -(\gamma-1)u_1u_3 & -(\gamma-1)u_2u_3 & a_{54} & \gamma u_3 \end{pmatrix}$$

$$a_{41} = (\frac{\gamma - 1}{2})u^2 - u_3^2$$

$$a_{51} = u_3\Big((\gamma - 1)u^2 - \gamma e\Big)$$

$$a_{54} = \gamma e - (\gamma - 1)(u_3^2 + u^2/2)$$

All the formulas now refer to V. The following combinations of variables are introduced to simplify subsequent writing:

$$\bar{\gamma} = \gamma - 1, \qquad k_1 = \frac{V_2^2 + V_3^2 + V_4^2}{2V_5}, \qquad k_2 = k_1 - \gamma$$

$$k_3 = k_1^2 - 2\gamma k_1 + \gamma, \qquad k_4 = k_2 - \bar{\gamma}, \qquad k_5 = k_2^2 - \bar{\gamma}(k_1 + k_2)$$

$$c_1 = \bar{\gamma}V_5 - V_2^2, \qquad d_1 = -V_2V_3, \qquad e_1 = V_2V_5$$

$$c_2 = \bar{\gamma}V_5 - V_3^2, \qquad d_2 = -V_2V_4, \qquad e_2 = V_3V_5$$

$$c_3 = \bar{\gamma}V_5 - V_4^2, \qquad d_3 = -V_3V_4, \qquad e_3 = V_4V_5$$

The Euler fluxes may be written as:

$$F_1(V) = \frac{\rho\iota}{V_5}\begin{pmatrix} e_1 \\ c_1 \\ d_1 \\ d_2 \\ k_2V_2 \end{pmatrix}, \quad F_2(V) = \frac{\rho\iota}{V_5}\begin{pmatrix} e_2 \\ d_1 \\ c_2 \\ d_3 \\ k_2V_3 \end{pmatrix}, \quad F_3(V) = \frac{\rho\iota}{V_5}\begin{pmatrix} e_3 \\ d_2 \\ d_3 \\ c_3 \\ k_2V_4 \end{pmatrix}$$

The matrix A_0 and its inverse are given by

$$A_0 = U_{,V} = \frac{\rho\iota}{\bar{\gamma}V_5}\begin{pmatrix} -V_5^2 & e_1 & e_2 & e_3 & V_5(1-k_1) \\ & c_1 & d_1 & d_2 & V_2k_2 \\ & & c_2 & d_3 & V_3k_2 \\ & symm & & c_3 & V_4k_2 \\ & & & & -k_3 \end{pmatrix}$$

and

$$A_0^{-1} = V_{,U} = \frac{-1}{\rho_t V_5} \begin{pmatrix} k_1^2+\gamma & k_1 V_2 & k_1 V_3 & k_1 V_4 & (k_1+1)V_5 \\ & V_2^2 - V_5 & -d_1 & -d_2 & e_1 \\ & & V_3^2 - V_5 & -d_3 & e_2 \\ & symm & & V_4^2 - V_5 & e_3 \\ & & & & V_5^2 \end{pmatrix}$$

The Jacobians of the Euler fluxes are:

$$\tilde{A}_1 = F_{1,V} = \frac{\rho_t}{\bar{\gamma} V_5^2} \begin{pmatrix} e_1 V_5 & c_1 V_5 & d_1 V_5 & d_2 V_5 & k_2 e_1 \\ & -(c_1 + 2\bar{\gamma} V_5) V_2 & -c_1 V_3 & -c_1 V_4 & c_1 k_2 + \bar{\gamma} V_2^2 \\ & & -c_2 V_2 & -d_1 V_4 & k_4 d_1 \\ & symm & & -c_3 V_2 & k_4 d_2 \\ & & & & k_5 V_2 \end{pmatrix}$$

$$\tilde{A}_2 = F_{2,V} = \frac{\rho_t}{\bar{\gamma} V_5^2} \begin{pmatrix} e_2 V_5 & d_1 V_5 & c_2 V_5 & d_3 V_5 & k_2 e_2 \\ & -c_1 V_3 & -c_2 V_2 & -d_1 V_4 & k_4 d_1 \\ & & -(c_2 + 2\bar{\gamma} V_5) V_3 & -c_2 V_4 & c_2 k_2 + \bar{\gamma} V_3^2 \\ & symm & & -c_3 V_3 & k_4 d_3 \\ & & & & k_5 V_3 \end{pmatrix}$$

$$\tilde{A}_3 = F_{3,V} = \frac{\rho_t}{\bar{\gamma} V_5^2} \begin{pmatrix} e_3 V_5 & d_2 V_5 & d_3 V_5 & c_3 V_5 & k_2 e_3 \\ & -c_1 V_4 & -d_2 V_3 & -c_3 V_2 & k_4 d_2 \\ & & -c_2 V_4 & -c_3 V_3 & k_4 d_3 \\ & symm & & -(c_3 + 2\bar{\gamma} V_5) V_4 & c_3 k_2 + \bar{\gamma} V_4^2 \\ & & & & k_5 V_4 \end{pmatrix}$$

APPENDIX C

PETROV-GALERKIN OPERATORS FOR ADVECTION EQUATION

Chapter 2 described the Petrov-Galerkin formulation which uses the streamline, discontinuity capturing, and reduced discontinuity capturing operators to detail flow discontinuities. The need for these three operators can be illustrated by application of the Petrov-Galerkin formulation to model equations, such as the advection equation. The advection of u can be written as,

$$u_{,t} + \tilde{a}^T \nabla u = 0 \tag{C.1}$$

where $u = u(x,t)$ and $\tilde{a}$ is the characteristic vector. The simple Galerkin formulation results in the weighted residual equation given by,

$$\int_A W^T (u_{,t} + \tilde{a}^T \nabla u)\, dA = 0 \tag{C.2}$$

where $W = N$, the shape functions. For the finite element procedure the use of the Galerkin procedure is equivalent to a central differencing of the spatial derivatives which results in oscillatory solutions. Diffusion needs to be added to the scheme to suppress these oscillations. The direction of the added diffusion can be obtained from the characteristic vector $\tilde{a}$ which defines the direction of propagation of information. A way to add this diffusion is to modify the interpolation functions W by adding a discontinuous function p to N. The

function p can be written as,

$$p = \tau \, \tilde{a}^T \nabla N \tag{C.3}$$

where τ depends on the characteristic vector $\tilde{a}$. The use of the characteristic vector (and the use of the characteristic matrices $\tilde{A}_i$ for the compressible flow equations) injects into the finite element formulation the eigenvalue/eigenvector information of the hyperbolic system. The addition of p to the weighting functions can be shown to result in the addition of artificial dissipation terms to the advection equations. To illustrate this, consider the weighted residual statement of Eq. (C.2). With the definition of W as,

$$W = N + p \tag{C.4}$$

Eq. (C.2) can be written as,

$$\int_A (N + p)^T \, [u_{,t} + \tilde{a}^T \nabla u] \, dA = 0 \tag{C.5}$$

or

$$\int_A N^T \, (u_{,t} + \tilde{a}^T \nabla u) \, dA = \int_A p^T \, (u_{,t} + \tilde{a}^T \nabla u) \, dA \tag{C.6}$$

Using the discontinuous function p for the spatial derivatives only, Eq. (C.6) can be written as,

$$\int_A N^T \, (u_{,t} + \tilde{a}^T \nabla u) \, dA = \int_A p^T \, \tilde{a}^T \nabla u \, dA \tag{C.7}$$

or

$$\int_A N^T \, (u_{,t} + \tilde{a}^T \nabla u) \, dA = \int_A (\tau \tilde{a}^T \nabla N)^T \tilde{a}^T \nabla u \, dA \tag{C.8}$$

Using integration by parts, Eq. (C.8) can be written as,

$$\int_A N^T \, (u_{,t} + \tilde{a}^T \nabla u) \, dA = \int_A N^T \nabla (\tilde{a} \tau \tilde{a}^T \nabla u) \, dA \tag{C.9}$$

which is the weighted residual statement of Eq. (C.1) with added artificial dissipation.

The addition of the streamline diffusion terms enhances the ability of the Petrov-Galerkin formulation to capture discontinuities but the oscillations at these discontinuities are not completely eliminated. Additional diffusion needs to be added at the shocks. The direction of the streamlines is along $\tilde{a}$ while the direction of the discontinuity is grad(u) and these directions are shown in Fig. C.1. To capture discontinuities better, diffusion needs to be added normal to the discontinuity or along grad(u).

To get the projection of $\tilde{a}$ onto the direction normal to the discontinuity, $\tilde{a}$ can be split into components in directions normal and parallel to the discontinuity as shown in Fig. C.1.

$$\tilde{a} = \tilde{a}_{\parallel} + \tilde{a}_{\perp} \tag{C.10}$$

By definition, $\tilde{a}$ satisfies the relation

$$\tilde{a}_{\parallel}^{T} \nabla u = \tilde{a}^{T} \nabla u \tag{C.11}$$

The addition of diffusion normal to the discontinuity is enabled using $\tilde{a}_{\parallel}$ which contains this directional information. The addition of the required dissipation can be accomplished by modifying the weighting function further to include the "discontinuity capturing" term. W is modified as

$$W = N + p + q \tag{C.12}$$

where q can be written as,

$$q = \tau_{\parallel} \tilde{a}_{\parallel}^{T} \nabla N \tag{C.13}$$

The addition of q to the weighting function adds diffusive terms on the

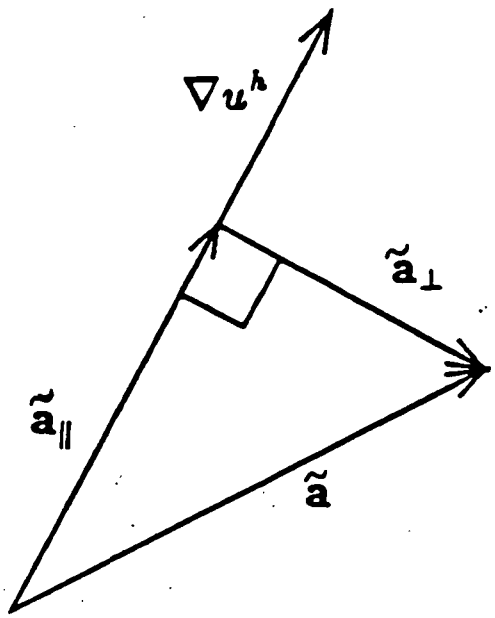


Fig. C.1 Decomposition of $\tilde{\mathbf{a}}$ into parallel and normal components.

right hand side similar to the streamline term and this is given by,

$$\int_A N^T (u,_t + \tilde{a}^T \nabla u)\, dA = \int_A N^T \nabla(\tilde{a}\, \tau \tilde{a}^T \nabla u)\, dA + \int_A N^T \nabla(\tilde{a}_{\parallel}\, \tau_{\parallel} \tilde{a}^T \nabla u)\, dA \tag{C.14}$$

With the contributions of p and q to the weighting function W, too much diffusion is added normal to the discontinuity. The diffusion needed at the discontinuity can be explicitly controlled by q which implies that the component of the streamline operator along the direction of grad(u) is redundant and can be subtracted out. This avoids the occurrence of overly smoothed discontinuities.

The use of streamline, discontinuity capturing and reduced discontinuity capturing terms in the weighting function serves to add artificial diffusion in a manner similar to the explicit viscosity schemes. The use of the characteristic vector and its projection in the direction of the discontinuity provide control of the direction of the added diffusion. The directional addition of diffusion serves to provide accurate details of flow discontinuities.